Estimating Dynamic Traffic Matrices by Using Viable Routing Changes

Augustin Soule, Antonio Nucci, *Senior Member, IEEE*, Rene L. Cruz, *Fellow, IEEE*, Emilio Leonardi, *Member, IEEE*, and Nina Taft, *Member, IEEE*

***Abstract*—In this paper we propose a new approach for dealing with the ill-posed nature of traffic matrix estimation. We present three solution enhancers: an algorithm for deliberately changing link weights to obtain additional information that can make the underlying linear system full rank; a cyclo-stationary model to capture both long-term and short-term traffic variability, and a method for estimating the variance of origin-destination (OD) flows. We show how these three elements can be combined into a comprehensive traffic matrix estimation procedure that dramatically reduces the errors compared to existing methods. We demonstrate that our variance estimates can be used to identify the elephant OD flows, and we thus propose a variant of our algorithm that addresses the problem of estimating only the heavy flows in a traffic matrix. One of our key findings is that by focusing only on heavy flows, we can simplify the measurement and estimation procedure so as to render it more practical. Although there is a tradeoff between practicality and accuracy, we find that increasing the rank is so helpful that we can nevertheless keep the average errors consistently below the 10% carrier target error rate. We validate the effectiveness of our methodology and the intuition behind it using commercial traffic matrix data from Sprint's Tier-1 backbone.**

***Index Terms*—Network tomography, SNMP, traffic engineering, traffic matrix estimation.**

I. Introduction

A TRAFFIC MATRIX is a representation of the volume of traffic that flows between origin-destination (OD) node pairs in a network. In the context of the Internet, the nodes can represent Points-of-Presence (PoPs), routers or links. In current IP backbone networks, obtaining accurate estimates of traffic matrices is problematic. There are a number of important traffic engineering tasks that could be greatly improved with the knowledge provided by traffic matrices. As a result, network operators have identified a need for the development of practical methods to obtain accurate estimates of traffic matrices. Example applications of traffic matrix estimation include logical topology design, capacity planning and forecasting, routing protocol configuration, provisioning for Service Level Agreement (SLAs), load balancing, and fault diagnosis.

Direct measurement of traffic matrices across an entire ISP network is considered too cumbersome in terms of communication and computation overhead to be practical with the current state of the art flow monitoring equipment [7]. Research in this area has thus turned to statistical inference techniques. The basic problem these techniques are applied to is the following. The relationship between the traffic matrix, the routing and the link counts can be described by a system of linear equations $Y = AX$, where Y is the vector of link counts, X is the traffic matrix organized as a vector, and A denotes a routing matrix in which element a_{ij} is equal to 1 if OD pair j traverses link i or zero otherwise. The elements $\{a_{ij}\}$ take on values {0,1} when traffic splitting is not allowed, but can take on fractional values when traffic splitting is supported. In networking environments today, Y and A are readily available; the link counts Y can be obtained through standard SNMP measurements and the routing matrix A can be obtained by examining IGP link weights together with the corresponding topological information. The problem at hand is to estimate the traffic matrix X. This is not straightforward because the matrix A does not have full rank, and hence the fundamental problem is that of a highly under-constrained, or ill-posed, system.

A first generation of techniques were proposed in [1]–[3]. Model parameters are estimated using either Moment Generating methods [1], Bayesian methods [2] or Maximum Likelihood estimation [3]. A common idea behind these approaches to tackle the highly under-constrained problem was to introduce additional constraints related to the second order moment of the OD pairs. Estimation is then carried out with two batches of constraints, one on the first order moment and one batch for the second order moment. However, the combined set of constraints is not solvable without an assumption on the relationship between the mean and variance. For example, in [1], [2] the authors assume that the volume of traffic for a given OD pair has a Poisson distribution, thus with an identical mean and variance. Cao *et al.* [3] assume instead that the traffic volume for OD pairs follows a Gaussian distribution, and that a power law relationship between the mean and variance exists. A comparative study of these methods [4] revealed that these methods were highly dependent upon the initial starting point, often called a *prior*, of the estimation procedure. Hence a second generation of techniques emerged [4]–[6] that proposed various methods for generating intelligent priors.

In practice, all of these statistical inference techniques for estimating traffic matrices suffer from limited accuracy. The magnitude of estimation errors are distributed over a range, however,

Manuscript received August 31, 2004; approved by IEEE/ACM Transactions on Networking Editor D. Veitch. This work was done when A. Nucci and N. Taft were at Sprint Labs.

A. Soule is with Thomson Research, Paris, France.
A. Nucci is with Narus Inc., Mountain View, CA 94043 USA (e-mail: anucci@narus.com).
R. L. Cruz is with the University of California at San Diego, La Jolla, CA 92093 USA.
E. Leonardi is with the Dipartimento di Elettronica, Politecnico di Torino, 10129 Turin, Italy.
N. Taft is with Intel Research, Berkeley, CA 94704 USA.

Digital Object Identifier 10.1109/TNET.2007.893227

in short we can say that the errors typically fall between 10 and 25%. Also these methods yield outliers such that some OD pairs can have errors above 100%. It has been difficult to drive the estimation error down below these values. Some carriers have indicated that they would not use traffic matrices for traffic engineering unless the inference methods could drive the *average* errors below the 10% target.

The fundamental difficulty in the traffic matrix (TM) estimation problem is due to it being under-constrained. The natural question to ask is whether or not there is anything we can do to increase the rank of the underlying linear system. In this work we propose the idea of deliberating changing the link weights so as to alter the shortest path routes used between ingress and egress nodes. The idea is by that doing this and collecting new SNMP link measurements under these altered routing states, we collect more measurements that are linearly independent of the existing measurements, thereby increasing the rank of the basic system $Y = AX$. In order to do this, an algorithm is needed to select which weights to change and by how much their values should be altered [9].

Such an algorithm cannot be used alone to create a solution to the TM problem because it would lead to a solution that is neither complete nor practical. The solution is not complete because the routing configuration changes would have to be applied over a multi-hour period (discussed below). The traffic itself during such time periods is non-stationary and thus the underlying traffic model used in the inference solution needs to be able to capture long-term traffic behaviors. The solution may not be practical because carriers want to conduct a limited number of such routing configuration changes to avoid further complicating the management of commercial networks. In this paper, we develop a methodology for traffic matrix estimation solution that resolves both of these problems.

Our methodology leads to two algorithms, the first allows one to estimate *all* the OD flows in a traffic matrix with high accuracy by combining cyclo-stationary models with routing changes. The second algorithm yields a method for estimating only the elephant flows in a traffic matrix. This can be done with very few routing changes thus rendering the method practical while maintaining good accuracy. A comparison of these two methods illustrates the tradeoff between practicality (limited routing changes) and accuracy.

Our contributions are multiple.

• We propose the idea of changing routing configurations in order to increase the rank of the linear system used for traffic matrix estimation. This reduces the inherent ill-posed nature of the basic problem.

• We provide an identifiability result for a TM estimation method that incorporates route changes. We prove that the first order moment of the OD flows is always identifiable under a proper sequence of routing configuration changes for any bidirectional biconnected topology. In other words, it is always possible to obtain a full rank system for such a topology.

• We develop a new model for OD flows that captures their behavior over long time periods (e.g., multiple hours or days). This enables traffic matrix estimation methods that may take a long time to collect all the data needed to estimate accurately. Our OD flow model contains two critical components, diurnal patterns that capture (long-term) cyclo-stationary behavior and a fluctuations process capturing short-term variability.

• We derive a closed form solution to estimate the covariance of the TM based on our underlying OD models. To the best of our knowledge, this is the first time that an estimate for the covariance of a TM has been proposed. We also prove that under certain conditions the variance will always be identifiable, i.e., that a unique solution exists.

• We explain how to incorporate our three solution enhancements (route changes, cyclo-stationary models and variance estimators) into a complete traffic matrix estimation methodology. Depending upon how these enhancements are combined, our methodology yields two different solutions, one for estimating the entire traffic matrix and one for estimating the elephant OD flows only. By switching the goal (and solution) to focus only on heavy OD pairs we render our solutions more practical. This is because with fewer variables to estimate, fewer weight changes are needed and less data needs to be collected. By incorporating the well accepted idea, that tiny flows are unimportant for most traffic engineering tasks into our methods, we show how TM estimation can be simplified while retaining accurate estimates of the heavy flows.

• We propose the first method that does not require an assumption about the relationship between the mean and the variance of OD flows.

With our methods we can drive the average error rates down into a whole new range, namely below the 10% target; and often we reach 4 or 5% average error rates (depending upon the scenario). To the best of our knowledge, this is the first paper that consistently achieves errors below this 10% barrier.

Our composite solution to the traffic matrix estimation problem involves many elements. Part 1 of our solution was presented in [9] where we first proposed the idea of changing link weights so as to increase the rank of the system. In that paper we showed that the problem of finding a *minimal* set of link weight changes to achieve full rank is NP-hard and we provided a heuristic algorithm for determining which links are most advantageous to change and by how much a weight value should be altered. Part 2 of our solution appeared in [10], that presents our cyclo-stationary models, the variance estimator, and explains how the models and routing changes are incorporated into a traffic matrix estimation procedure. In this journal version of our work, we present (in a single paper) the entire methodology (derived from the previous two papers), with a more comprehensive perspective. To do this we include all the steps of the methods, including the Viable Routing Changes algorithm (VRC-heuristic), for completeness. However, we have removed some discussions on subtle points, examples illustrating the effect of steps in the weight change selection algorithm, and so on. This journal version also differs from the previous two papers in that it contains our result on identifiability of the first order moment, which has never been presented before.

II. Problem Statement

The network traffic demand estimation problem can be formulated as follows. Consider a network represented by a collection $\mathcal{V} = \{1, 2, \ldots, V\}$ of nodes, and a set of L directed links $\mathcal{L} \subset \mathcal{V} \times \mathcal{V}$. Each node represents a set of co-located routers (PoP). Each link represents an aggregate of transmission resources between two PoPs. (We assume there are no self directed links, e.g., there are no links l of the form $l = (i, i)$.) We consider a finite time horizon consisting of K disjoint measurement intervals, indexed by $k \in [0, \ldots, K-1]$. We refer to each measurement interval as a **snapshot**. In each snapshot we change the link weights, i.e., the paths followed for some OD pairs, and this gives a different image of OD pairs traversing the network. We assume the routing remains unchanged within the same snapshot, while two different snapshots are characterized by two different routing scenarios. Within each snapshot, we collect multiple consecutive readings of the link counts using the SNMP protocol at discrete time n, indexed from 0 to $N_s - 1$. Each reading constitutes one **sample**. Each snapshot lasts for $N_s \times 5$ minutes because SNMP reports link counts every 5 minutes.

Consider an origin destination (OD) pair $p = (v_1, v_2)$, with $v_1 \neq v_2$, where v_1 denotes a traffic source (ingress node) and v_2 denotes a traffic sink (egress node). Let $X_p(k, n)$ be the amount of traffic associated with OD pair p (i.e., originating at node v_1 and departing the network at node v_2) during measurement interval k at discrete time n. We assume that the measurement intervals are long enough so we can ignore any traffic stored in the network. Let $\mathcal{P}$ denote the set of all OD pairs; there are a total of $|\mathcal{P}| = P = V^2 - V$ OD pairs.[1] We order the OD pairs p and form a column vector $X(k, n)$ whose components are $X_p(k, n)$ in some pre-defined order.

Let $Y_l(k, n)$ be the total volume of traffic that crosses link $l \in \mathcal{L}$ during measurement interval k at time n. $Y(k, n)$ represents the column vector whose components are $Y_l(k, n)$ for all $l \in \mathcal{L}$. Let $A_{l,p}(k)$ be the fraction of the traffic $X_p(k, n)$ from OD pair p that traverses link l during measurement interval k at time n. Thus, $Y_l(k, n) = \sum_{p \in \mathcal{P}} A_{l,p}(k) X_p(k, n)$. Forming the $L \times P$ matrix $A(k)$ with elements $\{A_{l,p}(k)\}$, we have in matrix notation

$$Y(k, n) = A(k)X(k, n) \ \forall k \in [0, K-1] \text{ and } n \in [0, N_s - 1]. \quad (1)$$

In the literature, the $Y(k, n)$ vector is called the *link count vector*, while $A(k)$ is called the *routing matrix*. In IP networks, the routing matrix $A(k)$ during each measurement interval k can be obtained by gathering topological information, as well as OSPF or ISIS link weights. Link counts $Y(k, n)$ are obtained from SNMP data.

A general problem considered in the literature is to compute an estimate of the traffic matrix $X(k, n)$ for each k and n given the observed link count vectors $Y(k, n)$ for each k and n, assuming that the routing matrix $A(k)$ does not change in time, i.e., $A(k_1) = A(k_2) \forall k \in [0, K-1]$, and that $X(k, n)$ is a sample of a stationary (vector valued) random process. Furthermore it is generally assumed in prior work that the components of $X(k, n)$ are uncorrelated. The general problem is non-trivial since the rank of $A(k)$ is at most L and $L \ll P$, i.e., for each k the system of equations (1) is under-determined.

[1]In our analysis we neglect local traffic, i.e., traffic that enters and leaves the network at the same node, since it does not contribute to the load on any inter-PoP link.

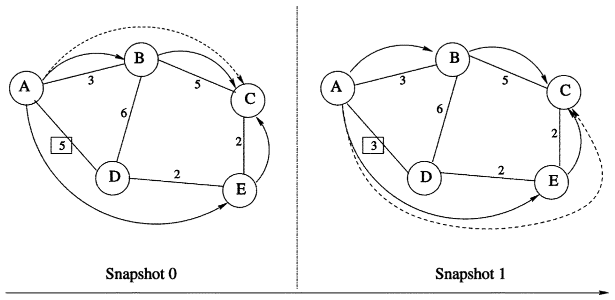

Fig. 1. Example: impact of routing changes.

The problem we consider in this paper is broader because we will allow the routing to change, so that we may indeed have $A(k_1) \neq A(k_2)$, and we will consider the case when $X(k, n)$ does not come from a stationary process.

III. Route Changes

We now explain, via an example, why the idea of changing the routing can help increase the rank of the system.

A. *Essence of the Idea*

Consider the network shown in Fig. 1 composed of five nodes interconnected by six unidirectional links. Each link has an associated weight and the traffic from each OD pair is routed along the shortest cost path. For simplicity, we consider only five OD pairs (indicated by arrows). On the left of Fig. 1 we represent the network in its normal state, when no link weight changes have been effected. Snapshot 0 would generate the following system of linear equations:

$$\begin{bmatrix} Y_{AB} \\ Y_{AD} \\ Y_{BD} \\ Y_{BC} \\ Y_{EC} \\ Y_{DE} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} X_{AB} \\ X_{AC} \\ X_{AE} \\ X_{BC} \\ X_{EC} \end{bmatrix}$$

The rank of routing matrix $A(0)$ is four. Two of the five OD pairs ((A, E) and (E, C)) can be estimated exactly because in this simple example they do not share their links with other OD pairs. On the right of Fig. 1 we show the effect of decreasing the weight of link $l = (A, D)$ from 5 to 3 (snapshot $k = 1$). This perturbation in the weights causes the rerouting of the OD pair (A, C) through the new path $\{(A, D), (D, E), (E, C)\}$. This snapshot generates a new system of linear equations, i.e., a new routing matrix $A(1)$, that can be appended to the previous set. One line of the new routing matrix would look like $Y_{AB} = [1\ 0\ 0\ 0\ 0]\ [X_{AB}\ \ldots]^T$. We can see that adding this to the original system of equations adds a new linearly independent equation into the system. As a consequence, the new system $[A(0)^T\ A(1)^T]^T$ is full rank and all five OD pairs can be estimated exactly.

B. *Identifiability of First Order Moment*

In this section we prove that the first order moment problem is *always identifiable* under a proper sequence of routing configuration changes for *any bidirectional biconnected* topology, i.e., it is always possible to select a proper sequence of routing configurations to obtain a full rank aggregate routing matrix A.

We start considering a bidirectional ring topology:

Theorem 1: On *any bidirectional ring topology* the first order moment estimation problem can be made identifiable under a proper sequence of minimum cost routing configurations.

Proof: Consider a bidirectional ring with V nodes. Let the nodes be numbered from 0 to $V-1$. Let L_v^+ identify the unidirectional link from node v to node $|v+1|_V$,[2] while L_v^- be the unidirectional link from node v to node $|v-1|_V$.

We start by setting the link costs in the following way: $\pi(L_0^+) = \epsilon \ll 1$, $\pi(L_v^+) = M > V$, for $v > 0$, and on the reverse ring we set up all the link costs to 1, i.e., $\pi(L_v^-) = 1$. Under this routing configuration, the link L_0^+ is shared only by traffic flowing from node 0 to node 1; thus OD pair $0 \to 1$ is identifiable. In a similar way we can identify any OD pair $v \to |v+1|_V$, by setting the proper routing configuration.

As a second step, we apply a new routing configuration as follows: $\pi(L_0^+) = \pi(L_1^+) = \epsilon \ll 1$, $\pi(L_v^+) = M > V$, for $v > 1$, and $\pi(L_v^-) = 1$ for all the links belonging to the reverse ring. Under this new routing configuration, the only OD pairs sharing link L_0^+ are $0 \to 1$ and $0 \to 2$. However, since OD $0 \to 1$ was already identified this traffic configuration allows us to identify $0 \to 2$. In an analogous way all the $v \to |v+2|_V$ can be identified.

By iterating this procedure, its is clear that all the OD pairs can be well identified. Moreover, with a similar procedure it is possible to make identifiable each OD pair under the constraint that costs are assigned to undirected links, i.e., links sharing the same end-points *must* have the same link costs. Fig. 2 shows an example of two routing configurations that allow to identify the OD pairs $0 \to 1$ and $0 \to 2$. ■

We now generalize previous arguments to every bidirectional biconnected topology:

Theorem 2: On *any bidirectional biconnected topology* the first order moment estimation problem can be made identifiable under a proper sequence of minimum cost routing configurations.

Proof: Consider a V-node topology, and two generic nodes v_i and v_j; since the topology is biconnected there must exist two completely disjoint paths p_1 and p_2 from v_i to v_j; due to the bidirectionality of the topology the concatenation of p_1 and p_2 forms a bidirectional ring. Let us set the cost of all the incoming links to the ring (not belonging to the ring) to $H > V$, and all the cost of outgoing links from the ring to $\epsilon \ll 1$. We set the cost of all other links that are not in the ring to 1. In this way we isolate the ring from the entire topology and we can apply the same procedure explained previously to identify each OD pair whose origin and destination belong to such a ring. Then we can select two other nodes and repeat this procedure until all the OD pairs are identified.

A similar procedure can be applied for the more realistic scenario of a bidirectional biconnected topology under the constraint that costs are assigned to undirected links. ■

Note that on topologies which are not biconnected, the first order moment estimation problem can not, in general, be made identifiable by changing the routing. Consider a V-node bidirectional tree topology as a counter-example. In such a topology the number of links L is $V-1$. Clearly for each OD pair only one possible route exists. Let A be the resulting routing matrix for the only possible routing strategy on the tree topology. We have then $\mathrm{rank}(A) \le L = V-1 < P$.

[2]where $|.|_M$ denotes the module M operator.

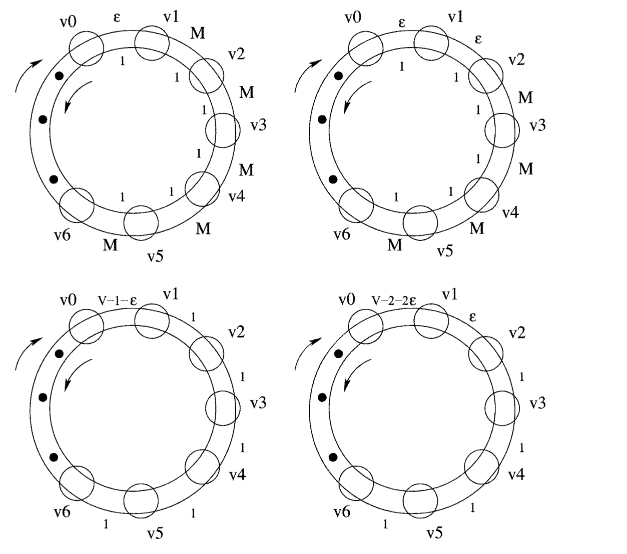

Fig. 2. Simple example showing how our procedure helps identify the first order moment of every OD pair, in a V-node bidirectional ring. On top the costs are assigned to each *directed link*. On the bottom the costs are assigned to each *undirected link*.

C. Practical Issues

To make use of this idea in a network, an algorithm is needed that can identify which link weights should be changed and by how much each weight should be altered. Our proposed algorithm, (presented in [9], and included in the Appendix herein for completeness) focuses on finding the *minimal* number of weight changes needed to reach full rank. We seek a minimal set of changes for the following reason. Consider the implications of the idea of changing link weights to increase the rank of the system. In practice, network operators cannot change link weights in rapid succession, it requires updating router configuration files across the entire network. Keeping the number of changes small thus eases the network management.

There is a second reason why we may need to wait an hour or more before switching between routing configurations. Once the weights have been changed, and the routing protocol has converged in computing new routes, we need to collect multiple samples of SNMP link counts under the new routing. It was shown in [3] that it is typically advantageous to use around 10 consecutive SNMP samples to take advantage of link correlations when making OD flow estimates. This implies 10 consecutive samples *under the same snapshot*. For these reasons it is likely to be a few hours (at least) between weight change events. In such time periods, the traffic is not stationary, and this has profound implications for the OD models used in TM estimation (addressed in Sections IV and VI).

IV. Traffic Dynamics

To understand traffic dynamics and what this may mean for OD traffic models, we begin by some exploratory analysis of our traffic matrix data. Our data includes one month of OD pair

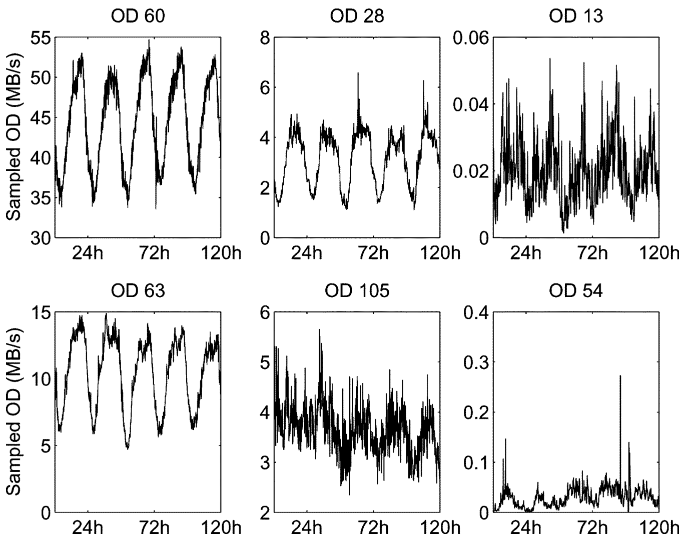

Fig. 3. Examples of representative OD flows: *large* on the left, *medium* on the middle and *small* on the right.

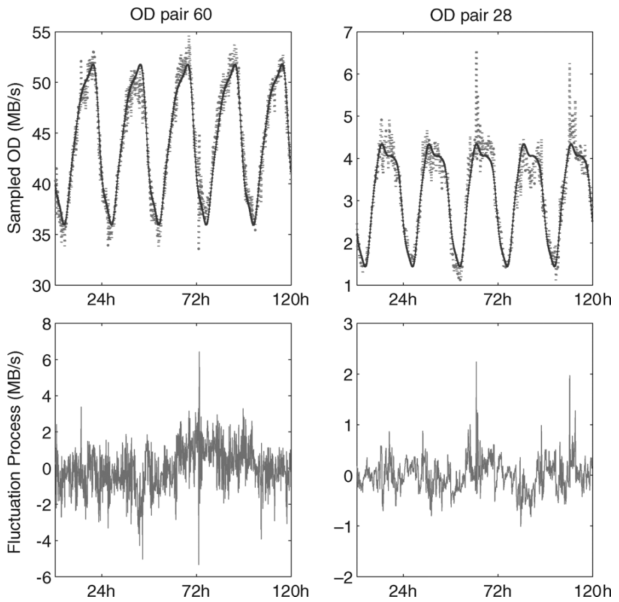

Fig. 4. Top: two example real large OD pairs (dotted lines) and their *diurnal patterns* (continuous lines). Bottom: the *fluctuation process* for the two OD pairs obtained by removing the diurnal trends from the original signals.

measurements collected in Sprint's IP backbone network by enabling Netflow on all the incoming links from gateway routers to backbone routers. The version of Netflow used (called *Aggregated Sampled Netflow*) deterministically samples 1 out of every 250 packets, and stores the 5-tuple from the sampled IP packet headers. Using local BGP tables and topology information we were able to determine the exit link for each incoming flow. The resulting link-by-link traffic matrix is aggregated to form both a router-to-router and a POP-to-POP traffic matrix.

Fig. 3 shows the evolution of six OD pairs in time across a week (excluding weekends). Here we show three types of behaviors and provide two OD pairs per type as illustrative examples. The two OD pairs on the left (top and bottom) are large and exhibit a regular cyclo-stationary behavior in time. There is a strong periodicity at intervals of 24 hours that fits the expected diurnal patterns. On the right we plot two OD pairs with extremely low mean rate (note the y-axes are different on each of these plots) and see that these flows are characterized by a very noisy shape. These OD pairs have mostly lost their cyclo-stationary shape and tend to be very bursty. The two OD pairs in the middle column, are in between these two behaviors: some retain small diurnal patterns while others become dominated by noisy behavior.

We examined our OD flows, ordered from largest to smallest by mean, and found that the heaviest 30% of our flows carry 95% of the total network traffic. This "elephant and mice" behavior has been observed countless times before in link traffic, web traffic and so on. It is not surprising that we find the same here in IP backbone OD flows. As we will see, focusing only on estimating the elephants flows makes the procedure easier to implement and thus more practical. Other traffic matrix studies have argued that the little flows (and errors in estimating them) can be ignored because network administrators only care about estimating the large flows correctly [5], [11]. The very small flows are not important because capacity planning tasks, route selection, load balancing, failure provisioning should be tailored to work for the vast majority of the traffic (put differently, for the heavy flows); but there is no need to optimize them for the last 5% of the load. In our data, selecting OD flows whose means are above a threshold of 3 MB/s, yields enough "elephant" or heavy flows so as to capture 95% of the total network load. We consider these flows in the latter portions of the paper that focus on estimating the heavy flows.

Fig. 3 hints that OD flows contain (at least) two sources of variability, namely *diurnal patterns* and a noisy *fluctuations behavior*. The figure shows these two behaviors across different flows. In fact, in the elephant flows both of these sources of variability often appear within each flow. To see this consider the two sample OD flows plotted in the top portion of Fig. 4. The real OD pairs are plotted with dotted lines. We used five basis functions of a Fourier series to filter out the diurnal patterns, represented by continuous lines. This represents the first component of an OD flow. The signal that remains after the diurnal pattern is filtered out is shown in the bottom plots. We call this latter component of an OD flow the fluctuations process.

We thus propose the following traffic model for origin-destination flows. *An OD flow model should contain two distinct components, one for the diurnal trend and one for the fluctuations process. The diurnal trend can be viewed as deterministic and cyclo-stationary, while the fluctuations process can be viewed as a stationary zero mean random process*. The stationarity of the fluctuations process is evidenced in the figure by a pretty consistent absence of any cycle trends over long time scales (hours and days). We said that one of our goals is to estimate the variance of the traffic matrix. In fact, what we will be estimating is the variance of this fluctuations process.

In the literature, it has been common to assume a fixed known relationship between the mean and standard deviation of an OD flow. The most common assumption is on the existence of a power law relationship between the two parameters. Fig. 5 shows the relationship between mean and standard deviation for our OD pair data sorted by their mean (from smallest to largest). The points on the plot do not approximate a straight

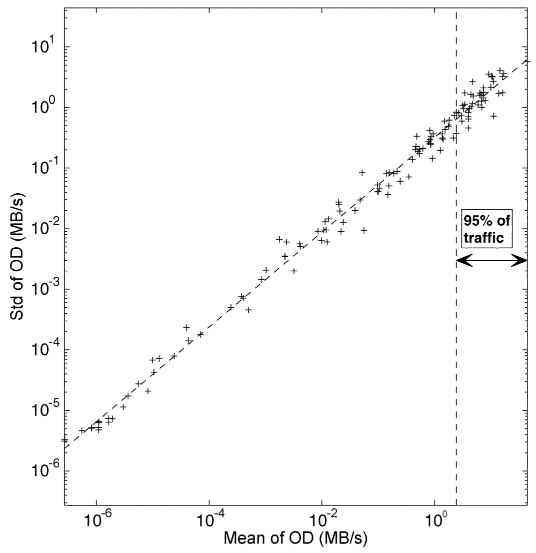

Fig. 5. Standard deviation of traffic fluctuations versus the mean traffic volumes for all the OD pairs (in Mbytes/s).

line as the assumption on the existence of a power law would entail. Note that this is a log-log plot, hence, the deviations from the straight line can be quite large. The best linear fitting for this data corresponds to a power law with exponent approximately equal to 0.78, that implies a variance power-law coefficient equal to 1.56. One of the difficulties with this assumption is that different researchers, with different datasets from different networks, each compute different values for this coefficient [3], [4]. Moreover, the coefficient varies per OD pair [4]. It is unclear how estimation errors are impacted by methods relying on this assumption.

In our work, we need not make any such assumption. Instead *we remove the power law assumption and choose to estimate the variance directly*, and do so independently of the mean. Being able to estimate the covariance of the TM has significant consequences: (1) we avoid having to make questionable assumptions about the relationship between the mean and variance; and (2) we can use our estimate to define a method for identifying the elephants flows. To see this, note that what Fig. 5 does confirm is the hypothesis that OD flows with large variance are also the ones with large mean. Hence there is an implication that the order of magnitude of the standard deviation is closely related to the order of magnitude of the mean for an OD flow. We will make use of this observation to help identify the top flows.

V. Methodology

Our composite methodology makes use of three new ideas: route changes, cyclo-stationary models and variance estimates. Before describing the details of our models and estimation procedures, we give here an overall summary of how these ideas are combined to build a composite methodology and some variants of it. In this paper we essentially propose two solutions, termed Algorithm 1 and 2, plus a variant of Algorithm 1. Algorithm 1 (below) can be used when one wants to estimate all the OD flows, while Algorithm 2 can be used if the one wants only to estimate the elephant OD flows.

Algorithm 1 Estimate *all* OD flows

```
1) Collect enough SNMP data to estimate the variance of
   OD flows.
2) Use VRC-heuristic to select minimal set of weight
   changes so as to generate full rank system.
3) Apply route changes and collect SNMP data from each
   snapshot.
4) Expand linear system to include all SNMP data and all
   routing configurations. process.
5) Use basic inference technique (pseudo-inverse or
   Gauss-Markov estimators) on full-rank expanded linear
   system.
```

We discuss what we mean by "enough SNMP data" in step 1 in Section VI-C. If one estimates the OD flow variances, then the Gauss-Markov estimator can be used in Step 5. A variant of this algorithm can be used if one does not have enough data to estimate the variances. In that case step 1 can be omitted and the pseudo-inverse estimator should be used in Step 5. The advantage of this variant is that it does not require any knowledge about traffic fluctuation statistics; however, the disadvantage is that it may be less accurate than the solution with the variance estimator. In the mathematical development of our methods, we also derive a closed form solution for evaluating the goodness of our variance estimate. This is included herein for completeness, although we do not evaluate it due to lack of space.

Algorithm 2 Estimate *heavy (elephant)* OD flows

```
1) Collect enough SNMP data to estimate the variance of
   OD flows.
2) Identify heavy flows. Use threshold policy (defined below)
   to select all OD flows whose variance is above threshold.
   Call the set $\mathcal{X}_L$. Reduce linear system by setting to zero
   all non-elephant flows.
3) Use VRC-heuristic to select minimal set of weight
   changes so as to make reduced system full rank.
4) Apply route changes and collect SNMP data from each
   snapshot.
5) Expand linear system to include all SNMP data and all
   routing configurations.
6) Use basic inference techniques (pseudo-inverse or
   Gauss-Markov estimators) on full-rank modified linear
   system.
```

A key component in Algorithm 2 is to identify the top largest OD flows (in advance of estimating their average volume). As observed earlier, a weak relation exists between the order or magnitude of the standard deviations and the order of magnitude of the means. By selecting the flows with the largest variance, we can be reasonably assured that we have identified the flows that are largest in mean. We then set the estimates for the small flows to zero and consider them known. We thus have fewer variables to estimate as only the large flows remain. Having a system with fewer unknowns, we can run our algorithm for finding the key link weight changes to increase the rank of the system. We

TABLE I
GLOSSARY OF NOTATION

Notation	Definition
L, l	number of network links, link index
P, p	number of OD pairs, OD pair index
K, k	number of measurement intervals (i.e., snapshots), snapshot index
N_s	number of SNMP samples collected in a snapshot
$Y_l(k,n)$	traffic volume (bytes) on link l during measurement interval k at discrete time n
$Y(k,n)$	link count vector
$X_p(k,n)$	traffic volume of OD pair p during measurement interval k at time n
$X(k,n)$	traffic matrix organized as a vector at time n during snapshot k
$A(k)$	routing matrix during snapshot k
A	block form of routing matrix including $A(k)$ $\forall k$
$W_p(k,n)$	fluctuation/noise process for OD pair p at time n in snapshot k
$W(k,n)$	noise column vector
t	time index, $t = kN_s + n$
C	block-form routing matrix relating OD fluctuations to link counts
B	covariance matrix of noise W
$r_p(t)$	autocorrelation of OD pair p over time lag t
$r(t)$	vector of OD pair autocorrelations. $r(0)$ is vector of OD variances
$R(t)$	matrix form with $r(t)$ on diagonal & 0's elsewhere
$C_Y(t)$	link correlation matrix
$\gamma_y(t)$	link covariance matrix ordered as a column vector
Γ	matrix whose rows are component-wise products of all pairs of rows of A
$b_h(\cdot)$,θ_h	basis function & coefficient of Fourier expansion
N_b	number of basis functions in cyclo-stationary model

expect to need fewer snapshots now since there are fewer variables to estimate.

Comparing the performance of these two solutions will illustrate an important tradeoff. In algorithm 1 we expect high estimation accuracy since the system is upgraded to full rank. However, this may require many snapshots (over 20). Algorithm 2 enables a smaller number of snapshots but potentially at the expense of accuracy in the estimates. Because the small flows are set to zero, their bandwidth will be redistributed to other flows, thus increasing estimation errors. This tradeoff is quantified in Section VII.

VI. Models and Estimates

Since we have an OD pair model that includes a diurnal pattern and a fluctuations process, to populate the traffic matrix with mean estimates now corresponds to extracting the main trend in time (i.e., a time-varying mean) of the OD pairs, while estimating the variance collapses to estimating the variance of the fluctuation process. (The notation introduced throughout this section is summarized in Table I.)

A. Incorporating Routing Changes

We now show how to incorporate the different $A(k)$ matrices (from different routing configurations) and all the SNMP data collected under each of the snapshots, into a expanded (block matrix) linear system. Our traffic matrix estimate here is for the variant of Algorithm 1 without the variance estimates. In order to build our solution step by step, we assume for the moment that $\{X(k,n) \forall k \in [0, K-1] \text{ and } n \in [0, N_s - 1]\}$ is a realization of a segment of a stationary discrete time random process. We model each OD pair as follows:

$$X(k,n) = x + W(k,n) \quad \forall k \in [0, K-1] \text{ and } n \in [0, N_s - 1] \tag{2}$$

where x is a deterministic column vector representing the mean of the OD pair, while $\{W(k,n)\}$ are zero mean column vectors, i.e., $E[W(k,n)] = \mathbf{0}$, representing the "traffic fluctuation" for the OD pairs at discrete time n in the measurement interval k. We define $W_p(k,n)$ as the p^{th} component of $W(k,n)$. Recall that the equation relating the OD traffic flows, routing, and link counts is as follows:

$$Y(k,n) = A(k)X(k,n) \ \forall k \in [0, K-1] \text{ and } n \in [0, N_s - 1]. \tag{3}$$

To incorporate all the SNMP samples from all of the snapshots, we build our expanded system by defining the following matrices, using block matrix notation.

$$Y = \begin{bmatrix} Y(0,0) \\ \vdots \\ Y(0,N_s-1) \\ \vdots \\ Y(K-1,N_s-1) \end{bmatrix}, \quad A = \begin{bmatrix} A(0) \\ \vdots \\ A(0) \\ \vdots \\ A(K-1) \end{bmatrix}$$

$$W = \begin{bmatrix} W(0,0) \\ \vdots \\ W(0,N_s) \\ \vdots \\ W(K-1,N_s-1) \end{bmatrix},$$

$$C = \begin{bmatrix} A(0) & 0 & \dots & 0 \\ 0 & A(0) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & A(K-1) \end{bmatrix}$$

In the new A matrix, each $A(k)$ matrix is repeated N_s times (for each SNMP sample), since the routing is assumed to stay the same within the same measurement interval k. We define M_k to be rank of matrix A_k. In terms of dimensions, Y is a LN_sK-dimensional column vector, A is a $LN_sK \times P$ dimensional matrix, W is a KN_sP-dimensional column vector, and C is a $LN_sK \times KN_sP$ matrix. Putting (3) into matrix form, using (2), we obtain

$$Y = Ax + CW. \tag{4}$$

Our aim is to estimate x, denoted $\hat{x}$, from the observations $Y(0,0), \dots, Y(0, N_s-1), \dots, Y(K-1, N_s-1)$. Throughout the remainder of the paper, we assume the VRC-heuristic is successful and that A is of full rank. One possible approach for synthesizing our estimate $\hat{x}$ is to minimize the Euclidean norm of $Y - Az$, i.e.,

$$\hat{x} = \arg\min_z \{(Y - Az)^T (Y - Az)\} \tag{5}$$

$$= (A^T A)^{-1} A^T Y. \tag{6}$$

This is the *Pseudo-Inverse* estimator, and does not require knowledge of the statistics of the traffic fluctuations W.

B. Incorporating Variance Estimates

We point out that the components of the link measurement vector Y will have different variances, due to the varying number of OD pairs traversing each link and the differences in the variance of the traffic fluctuations of individual OD pairs. Intuitively, a component of the link measurement vector should be weighted more heavily if it's variance is smaller. We were thus motivated to try to improve the above estimate by estimating (next subsection) and incorporating (this subsection) these variances. We now derive our estimate for the traffic matrix when the variances are known (for Algorithm 1). Let B be the covariance matrix of W, i.e.,

$$B = E[WW^T]. \tag{7}$$

Let t denote the discrete time across all the measurement intervals, from 0 to $T-1$ where $T=(KN_s)$ is the total number of samples collected across the whole experiment. There exists a bijective relationship between the discrete time index t and the pairs of temporal indexes k and n, i.e., $t=kN_s+n$, where $k=\lfloor t/N_s \rfloor$ and $n=t-\lfloor t/N_s \rfloor N_s$. Then the covariance matrix B can be written as:

$$B = \begin{bmatrix} R(0) & R(1) & \dots & R(T-1) \\ R(1) & R(0) & \dots & R(T-2) \\ \vdots & \vdots & \ddots & \vdots \\ R(T-1) & R(T-2) & \dots & R(0) \end{bmatrix} \tag{8}$$

where $R(t)$ is the $P \times P$ matrix defined by

$$R(t) = [r_p(t)] = \operatorname{diag}\left(E\left[W_p(\tau)W_p(\tau+t)\right]\right). \tag{9}$$

We assume in (9) that the traffic fluctuations across OD-pairs are uncorrelated, i.e., $E[W_p(\tau)W_{p'}(\tau+t)]=0$ if $p \neq p'$.

With this definition, the covariance matrix of Y is equal to CBC^T. Assuming (for now) that B is known, the best linear estimate of x given Y, in the sense of minimizing $E[(Y-Az)^T(Y-Az)]$ with respect to z is known as the best linear Minimum Mean Square Error (MMSE) estimator. The best linear MMSE estimate of x can be obtained from the Gauss-Markov Theorem [12], and is stated as

$$\hat{x} = \hat{x}(Y,B) = \left(A^T(CBC^T)^{-1}A\right)^{-1} A^T(CBC^T)^{-1}Y. \tag{10}$$

Note that the estimate $\hat{x}$ in (10) reduces to the pseudo-inverse estimate when CBC^T is the identity matrix. If W has a Gaussian distribution, it can be verified that the estimate in (10) is in fact the maximum likelihood estimate of x. This estimate has the following nice property:

Corollary 3: Regardless of whether or not W is Gaussian, the estimate in Proposition 10 is unbiased, i.e., $E[\hat{x}]=x$, and furthermore we have

$$\hat{x}(Y,B) = x + \left(A^T(CBC^T)^{-1}A\right)^{-1} A^T(CBC^T)^{-1}CW$$

and $E[(\hat{x}-x)(\hat{x}-x)^T] = (A^T(CBC^T)^{-1}A)^{-1}$.

The proof of this is straightforward and can be found in [10]. Note that this last equation allows us to estimate the accuracy of our estimate $\hat{x}$ of x given Y and B. In particular, the p^{th} element of the diagonal of the matrix $E[(\hat{x}-x)(\hat{x}-x)^T]$ is the mean square error of our estimate of the p^{th} element of the traffic matrix.

C. Estimating the Covariance of Traffic Fluctuations

We now present a method for obtaining an estimate of the covariance function of the fluctuations process $\{W\}$. We highlight two nice characteristics of our method. First, it does not require any knowledge of the first order moment statistics. Previous approaches assumed to know exactly the mean to recover the covariance and vice-versa. As a consequence, our method does not suffer from potential error propagation problems introduced by the first order moment estimation. Second, the estimate does not require any routing configuration changes. Instead, it relies on a large number of measurements of the link counts under the same routing configuration.

We define the **link correlation matrix** as $C_Y(t) = E[Y(\tau)Y^T(\tau+t)]$. For two links l and m, each entry of this matrix is given by

$$E\left[\sum_{i=1}^{P}\sum_{j=1}^{P} A_{l,i}\left[X_i + W_i(\tau)\right]A_{m,j}\left[X_j+W_j(\tau+t)\right]\right]$$
$$= \sum_{i=1}^{P} A_{l,i}A_{m,i}r_i(t) + \sum_{i=1,j=1,j\neq i}^{P} A_{l,i}A_{m,j}E[X_i]E[X_j].$$

In the previous statement we assumed that each OD pair is independent. By using a matrix notation, we can thus write

$$\begin{aligned} C_Y(t) &= AR(t)A^T + E[AX]E[AX]^T \\ &= AR(t)A^T + E\left[Y(\tau)\right]E\left[Y(\tau+t)\right]^T. \end{aligned} \tag{11}$$

The **link covariance matrix** is thus given by $C_Y(t) - E[Y(\tau)]E[Y(\tau+t)]^T$. This can be estimated directly from the sequence of link measurements contained in the link measurement vector Y. Once the link covariance matrix is known, we can estimate $R(t)$ as follows:

$$\hat{R}(t) = \arg\min_{Z}\left\|C_Y(t) - AZA^T - E[Y(\tau)]E[Y(\tau+t)]^T\right\|_2^2.$$

Notice that we can rewrite (11) as $\gamma_y(t) = \Gamma r(t)$, where $\gamma_y(t)$ is the link covariance matrix, $C_Y(t) - E[Y(\tau)]E[Y(\tau+t)]^T$, ordered as a L^2-dimensional column vector, Γ is a $L^2 \times P$ matrix whose rows are the component-wise products of each possible pair of rows from A, and $r(t)$ is a P-dimensional column vector whose elements are $r_p(t)$. With this formulation, an estimate of $r(t)$ can be obtained using the pseudo-inverse matrix approach:

$$\hat{r}(t) = (\Gamma^T\Gamma)^{-1}\Gamma^T\gamma_y(t). \tag{12}$$

Our estimate of the variance of the fluctuations process for each OD-pair is given by the components of $\hat{r}(0)$. Using a standard estimate for the link covariance matrix, we obtain

$$\hat{r}(0) = \frac{1}{T}\sum_{\tau=0}^{T-1}(\Gamma^T\Gamma)^{-1}\Gamma^T\hat{\gamma}_y(\tau) \tag{13}$$

where the components of $\hat{\gamma}_y(\tau)$ are $y_l(\tau)y_m(\tau) - \bar{y}_l\bar{y}_m$, and for each link l we define

$$\bar{y}_l = \frac{1}{T}\sum_{\tau=0}^{T-1} y_l(\tau).$$

To examine the accuracy of the estimate for $\hat{r}(t)$, we can derive the following after some algebraic manipulations:

$$\begin{aligned} E&\left[(\hat{r}(t)-r(t))^T(\hat{r}(t)-r(t))\right] \\ &\le \frac{1}{T^2}\|\Phi\|\sum_{i=0}^{T-1}\sum_{l=1}^{L}\sum_{m=1}^{L} E \\ &\quad\times\left[y_l^2(i)y_m^2(i+t) - \left(E\left[y_l(i)y_m(i+t)\right]\right)^2\right] \quad (14)\end{aligned}$$

where $\Phi = [(\Gamma^T\Gamma)^{-1}\Gamma^T]^T[(\Gamma^T\Gamma)^{-1}\Gamma^T]$, and $\|\Phi\|$ is the norm (i.e., determinant) of Φ. Equation (14) can be used to relate the confidence on the $r(t)$ estimate to the fourth order moments of link-count statistics, which can be evaluated through standard statistical techniques.

D. *Incorporating Cyclo-Stationarity*

Next, we present out cyclo-stationary model for traffic matrices and explain its impact on the associated estimation problem. We now forgo our previous assumption that $\{X(t)\}$ is a random process with constant mean as in (2). Instead we define our OD flow traffic with

$$X(t) = x(t) + W(t) \quad \forall t \in [0, T-1]. \quad (15)$$

where $\{X(t)\}$ is cyclo-stationary with period N—in the sense that $X(t)$ and $X(t+N)$ have the same marginal distribution. More specifically, we assume that $\{x(t)\}$ is a deterministic (vector valued) sequence, periodic with period N, and that $\{W(t)\}$ is a zero-mean stationary random process. In this framework, estimating the traffic matrix now corresponds to estimating $x(t)$ for all t given the observations $Y(t)$, $0 \le t < T = KN_s$.

We shall assume that $x(t)$ can be represented as the weighted sum of $2N_b + 1$ given basis functions, i.e.,

$$x(t) = \sum_{h=0}^{2N_b} \theta_h b_h(t) \quad (16)$$

where for each h, θ_h is a $P \times 1$ vector (of "coefficients"), and $b_h(\cdot)$ is a scalar "basis" function that is periodic with period N. In particular, we will consider a Fourier expansion where

$$b_h(t) = \begin{cases} \cos(2\pi th/N), & \text{if } 0 \le h \le N_b \\ \sin(2\pi t(h-N_b)/N), & \text{if } N_b+1 \le h \le 2N_b \end{cases} \quad (17)$$

Substitution of (16) into (15), and then (3), we obtain

$$\begin{aligned} Y(t) &= A(k)\left(\sum_{h=0}^{2N_b}\theta_h b_h(t)\right) + A(k)W(t) \\ &= A'(t)\theta + A(k)W(t) \end{aligned}$$

where we define the $(2N_b+1)P \times 1$ vector θ according to

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_{2N_b} \end{bmatrix} \quad (18)$$

and $A'(t)$ is the $L \times (2N_b+1)P$ matrix defined as

$$A'(t) = [A(k)b_0(t) \quad A(k)b_1(t) \quad \cdots \quad A(k)b_{2N_b}(t)]. \quad (19)$$

Next we redefine the matrix A to be of dimension $LT \times (2N_b+1)P$ as follows:

$$A = \begin{bmatrix} A'(0) \\ A'(1) \\ \vdots \\ A'(T-1) \end{bmatrix}. \quad (20)$$

The matrices C and W are defined as before. With this notation we have an equation similar to (4), namely

$$Y = A\theta + CW. \quad (21)$$

Thus, we can use the same estimate, from (10) as before, (or from (6)) to estimate θ when the covariance B is known (unknown), respectively. Then we estimate $x(t)$ for each instant t using (16). Our covariance matrix estimator in Section VI-C can still be used to obtain the covariances in the system described by (21). Our estimates for cyclo-stationary OD flows can be applied to the entire traffic matrix (Algorithm 1) or just the elephant flows (Algorithm 2).

E. *Identifiability of Second Order Moment*

In this section we prove that is always possible to estimate the covariance function without requiring any routing configuration changes for any topology.

Theorem: For a general connected topology the rank of Γ is P under any minimum cost routing in which link costs are strictly positive.

Proof: We will prove the theorem by contradiction. If the rank of Γ is smaller than P then $\ker\Gamma \ne 0$,[3] i.e., there exists a non-null vector $V_o \in R^P$ such that $\Gamma V_0 = 0$. Let $T \le P$ be the number of non-null components of V_0. Let $\mathcal{V}$ be the vector space of dimension T generated by all the vectors which have null components in correspondence to null components of V_0.

We consider the correspondence $F : \mathcal{V} \to R^T$ which maps any vector $V \in \mathcal{V}$ into a vector $W \in R^T$ by discarding null components of V. Let $\hat{\Gamma}$ be the matrix obtained by Γ by discarding the columns that in the multiplication ΓV with $V \in \mathcal{V}$ would not contribute (since multiplied by null elements of V). By construction $\Gamma V = \hat{\Gamma} W$ for any $V \in \mathcal{V}$. Finally let W_0 the vector which corresponds to V_0 though F. We notice that every component of W_0 is not null.

Let us consider the set Z_{od} of OD pairs which correspond to the elements of W_0. We will show that there exists at least a pair of links in the network which are jointly crossed by just one OD pair in Z_{od}. Thus, the row of $\hat{\Gamma}$ which corresponds to the

[3]We recall that, given a linear operator Γ, $\ker\Gamma$ denotes the set of vectors whose image through Γ is 0, i.e., $v \in \ker\Gamma$ iff $\Gamma v = 0$.

considered pair of links must contain only one element different from 0. As a consequence, necessarily, it results that $\hat{\Gamma} W_0 \neq 0$, in contradiction with the previous assumptions.

Indeed, let us compute path-cost (sum of the link weights) for any OD pair in Z_{od}. Let z_{od} be an OD pair in Z_{od} which corresponds to a maximum path cost. Consider the first and the last link (l and m respectively) spanned by the z_{od} path.

We claim that z_{od} is the only OD pair in Z_{od} which crosses both links l and m. By construction z_{od} crosses both l and m. In addition we show by contradiction that no other OD pairs can cross l and m. Assume $z'_{od} \neq z_{od}$ crosses both links l and m then there are only two possibilities: 1) l is the first and m the last link of the z'_{od} path; in this case, however, both the origin and the destination of z'_{od} would be coincident with the origin and the destination of z_{od} then contradicting the fact that $z'_{od} \neq z_{od}$; 2) either l is not the first link or m is not the last link of the path spanned by z'_{od}; in this case, however, the path cost of z'_{od} is larger than the path cost of z_{od} (since the sub-path of z'_{od} from l to m necessarily has the same cost of the whole z_{od} path) thus contradicting the fact that the path cost of z_{od} is maximum. ■

VII. Results

We now evaluate our two algorithms using the traffic matrix data from Sprint's commercial Tier-1 backbone. In both solutions, we used a pseudo-inverse estimator as our basic inference method in the last step. We used this, rather than the Gauss-Markov estimator because (as explained later), we did not have enough months of Netflow data to do a proper comparison using Gauss-Markov estimators.

1) Evaluation of Algorithm 1: Estimating the Whole Traffic Matrix: We now assess the accuracy of the mean estimation when the traffic matrix is computed using Algorithm 1. For our network scenario considered, the VRC-heuristic algorithm determined that $K = 24$ snapshots were sufficient to identify all the OD pairs. Of these 24 snapshots, 22 of them involve only one link weight change at a time, while the last two involve two simultaneous link weight changes.

For illustrative purposes, we first look at our estimates over time for six particular OD pairs (see Fig. 6). For each flow, these graphs show the temporal shapes of the real, de-noised, and estimated OD pair. The de-noised OD pair refers to an OD pair with everything filtered out except the first 5 basis functions of Fourier series; put alternatively, this illustrates how well a simple Fourier model captures the changing mean behavior. Our model fits these flows extremely well. It is interesting that the quality of the estimation obtained decreases as the average rate drops. The two on the right do suffer from larger errors. Note that these two OD flows (#28 and #105) were the worst case performing OD flow estimates from within the heavy flow category. Our method exhibits the same behavior as other methods in that it estimates large flows well and has difficulty as the flows get smaller and smaller.

The gain of our method comes in terms of the actual estimation errors achieved for these top flows. To examine estimation errors in general, we use the following two metrics. First, we examine the difference between the {\sl first component} of the Fourier series, i.e., the continuous component, for the Netflow data and the estimation provided by our models. Since this part

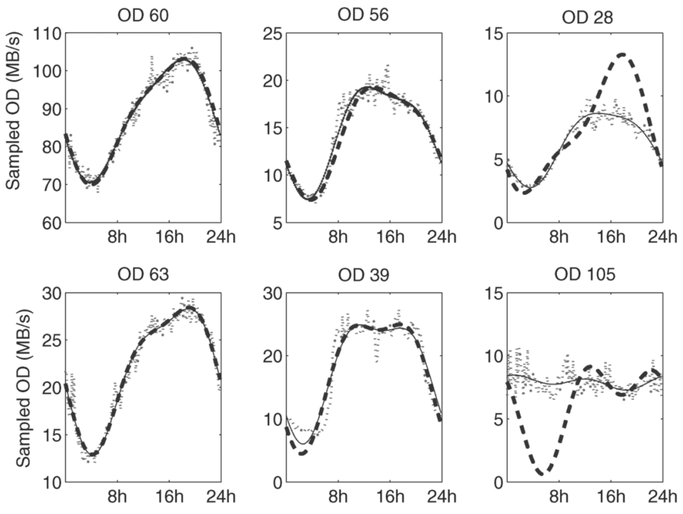

Fig. 6. Method 1. Mean Estimation. Real OD flow (continuous line), de-noised OD flow (dotted line), and estimated OD flow (dashed line).

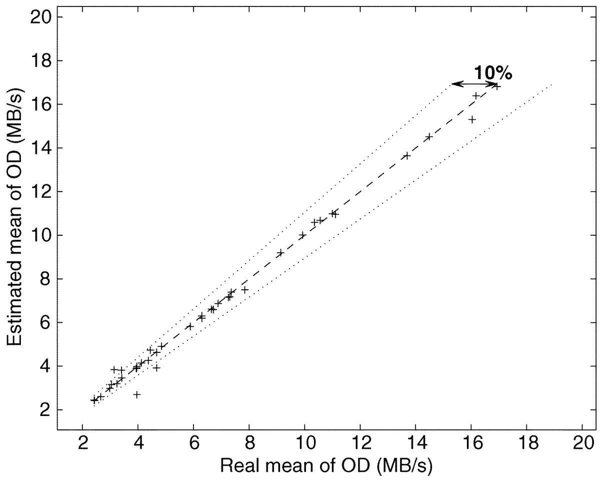

Fig. 7. Method 1. Relative error of 1st component of Fourier series for top flows with $N_s = 100$.

of our estimate would be used to populate a traffic matrix, we compute the relative error of this estimate. Second, since we have a temporal process, we need to examine the errors in estimation over time. We thus use the difference in energy between the estimate $\hat{X}_p(t)$ and the real data $X_p(t)$. In particular, we use the relative L2-norm to estimate the goodness of the model fitting, $||X_p(t) - \hat{X}_p(t)||_2^2 / ||X_p(t)||_2^2$.

Fig. 7 shows our first metric on the difference of the first component of the Fourier series for the real data (x-axis) and the estimated OD pairs (y-axis) for the elephant flows when $N_s = 100$ samples per snapshot are used. The average error is 3.8%. This is a large improvement over other methods whose *average* errors typically lie somewhere between 11%–23%.[4] Some carriers have indicated that they would not use traffic matrices for traffic engineering unless the inference methods could drive the average errors below the 10% barrier. We believe that this is the first study to achieve this.

[4]It is hard to compare numbers exactly because different studies use different amounts of total load. However, we capture more total traffic than most other studies that typically include 75% or 80% of the network-wide load.

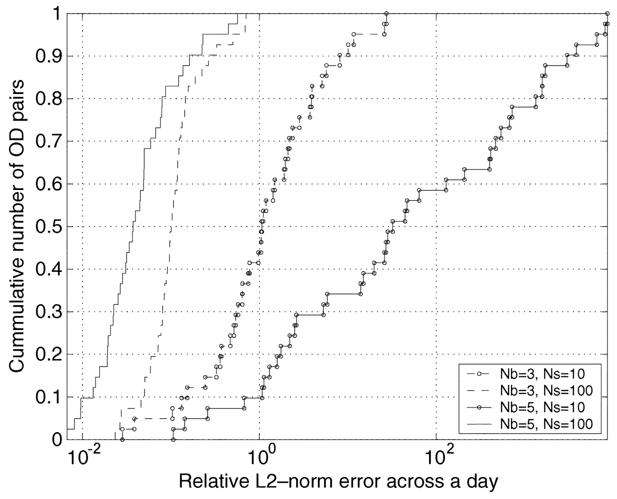

Fig. 8. Method 2. Cumulative distribution of the relative *L2-norm* error for top flows as a function of N_b and N_s.

Among these flows, the worst case relative error across all OD pairs is 28%. If we look carefully at the figure, we can see that all the flows have less than 10% relative error, except for four outliers. These outliers correspond to OD pairs whose average rate is less than 5 MB/s, i.e., among the smallest OD flows within the "heavy" ones represented here. For 90% of these top flows (i.e., excluding these four outliers), the average error drops below 2% and the worst case relative error drops to 4.6%. (See the first column in Table II for 24 snapshots.) We remind the reader that small OD pairs (those whose rate is under 3 MB/s) do not appear in the figure but are considered in the system and are thus a part of the overall computation.

Fig. 8 shows the *L2-norm* relative error cumulatively. In most of our calculations the number of basis functions we used was $N_b = 5$ and the number of samples per snapshot used was $N_s = 100$. This figure shows that the worst case L2-norm relative error was 56% corresponding to OD pair #28 (the second worst case is 44%, corresponding to OD pair #105). We see that 85% of the flows had an L2-norm error of less than 10% (see OD pairs #56 and #39 as an example). Fig. 6 helps us understand the effects of the L2-norm error. As we can see our model captures the diurnal trends correctly for the heavy OD pairs, but can suffer with relatively "small" OD flows.

We highlight the difference between our two error metrics. Others have reported relative errors on their mean estimates where the means are computed over some specific interval (often fairly long). Our L2-norm relative error is an error rate on our estimation (or fitting) of the dynamic OD flow varying in time; put alternatively it summarizes "instantaneous" errors in OD flow modeling. We cannot compare the value of this latter metric to other studies because they have not tried to build a model capturing the temporal behavior.

The performance of our method is influenced by the number of samples per snapshot and the number of basis functions used (see Fig. 8). Intuitively, a larger number of basis function N_b will lead to a better quality estimate, although at the cost of a larger number of samples. The number of samples N_s plays an important role independently of the number of basis functions implemented. The more samples collected, the more is learned about the temporal evolution of each OD pair, and a better estimation can be provided. For a fixed number of basis functions, going from $N_s = 10$ to $N_s = 100$ yields a substantial improvement. In our experimentation, using more than 5 basis functions yielded insignificant gains and thus we decided to set $N_b = 5$ for the remainder of our evaluation.

2) Evaluation of Algorithm 2: Estimating Elephant OD Pairs Only: We start by estimating the variance (Step 1) of the OD flows using (13). For Step 2, we order the OD flows by size of variance. By relying upon our observation that flows that have large variance are also typically large in mean, the task now is to set a threshold and select the "heavy" flows above this threshold. Two issues arise when doing this.

As discussed in Section VI, the nice feature about our variance estimator is that it does not require any routing changes. However, a large number of SNMP samples are required to force the relative error of the variance estimate to be under 5%. We selected the 5% target arbitrarily. After some experimentation we found that roughly 25,000 samples were needed to achieve this target level of accuracy. In a real network this implies one needs about 3 months worth of SNMP data. In commercial networks obtaining this much historical data is not a problem as all ISPs store their SNMP data for multi-year periods. Although we had plenty of months (years) of *SNMP* data, we did not have 3 months of *Netflow* data available to us that would have been needed to do a complete validation of this method.

We therefore decided to use pseudo-real data for this evaluation. We call this pseudo-real data because it is generated based on a model fitted to actual data (but only one month's worth). To create sample OD flows, we filter out the noise from each of our sampled OD pairs and keep only the first five components of the matched Fourier series. We generate sample OD flows, over longer periods of time, using this Fourier series model to which we add a zero mean gaussian noise with a power-law variance whose coefficient is set to 1.56 (in accordance with the empirical data observed in Fig. 5). We route this traffic (according to the original A matrix, i.e., snapshot $k = 0$) and generate what would be the resulting SNMP link counts for the 3 month period under study. This last step is the same as the methodology presented in [6].

We compare our estimate of the standard deviation (std) to the standard deviation of the pseudo-real data in Fig. 9. The top plot is for the elephant flows, while the bottom plot includes the comparison for small flows. The variance estimate for the medium and large OD flows is quite good in that it achieves an average estimation error of less than 5%. As expected, it is harder to estimate the variance of the smaller OD pairs and we see the errors can span a large range. This challenge cannot be met by merely increasing the number of samples because it is due to the difference in order of magnitude of large and small OD pairs. As a consequence, a small error in the std-estimate of large OD pairs will be spread across multiple OD pairs causing large errors in the std-estimate of small OD pairs.

Without a method for extracting exactly the top 30% largest OD pairs from the rest, we rely on a simple threshold scheme for

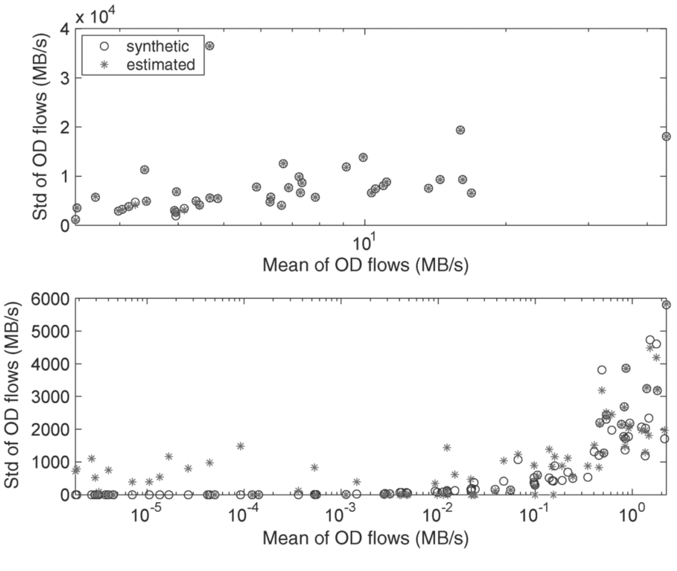

Fig. 9. Real and estimated standard deviation for all OD pairs, heavy (on the top) and small (on the bottom).

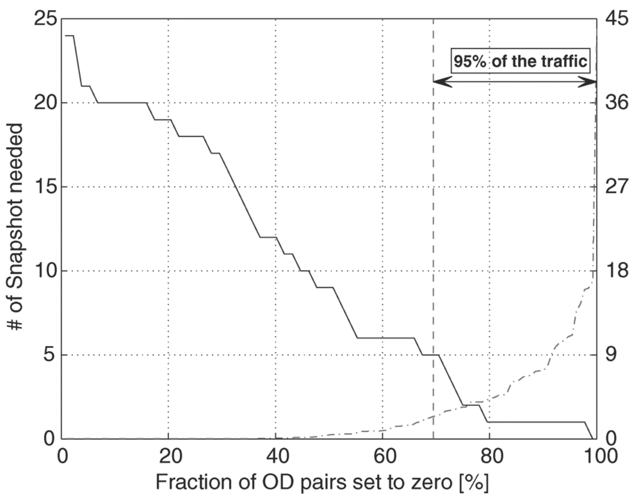

Fig. 10. Number of snapshots needed versus OD pairs eliminated.

now. We keep all OD pairs whose rates are above a threshold set to be two orders of magnitude lower than the largest OD pair. OD pairs below this threshold are set to zero.

We are now ready to examine the impact of removing the small OD pairs on the number of snapshots required. Having isolated the top OD flows, our methodology next uses a heuristic algorithm to select the sequence of snapshots needed—however, this time we seek identifiability on a much reduced set of OD pairs. Fig. 10 depicts the number of snapshots required as a function of the number of OD pairs set to 0. Recall that as more OD pairs get set to zero, there are fewer variables to estimate and so we expect the number of snapshots to be decreasing. We start with 24 snapshots, when all OD pairs are considered by the heuristic, and ending with 0 when no OD pair has to be identified. Note that only 5 snapshots are needed to disaggregate the 30% of largest OD pair carrying the 95% of network traffic. Recall that when we estimated the entire traffic matrix using Algorithm 1, our algorithm requires 24 routing configurations. Here we see that if we content ourselves with estimating 95% of the load of the traffic matrix, then we can drop the number of needed snapshots to 5, i.e., an 80% reduction! This is indeed a dramatic decrease in the number of needed snapshots. With the number of snapshots so small, ISPs are more likely to be able to execute such a method. One of our key findings here is thus that by focusing only on large flows and being able to identify them, we can enable a method (the weight change method) that appears to be able to significantly improve the error rates as compared to previous methods.

TABLE II
AVERAGE (ME) AND WORST-CASE (WC) RELATIVE ERRORS OF 1ST FOURIER COMPONENT FOR 100%, 95% AND 90% OF HEAVY OD PAIRS

OD [%]	**Snapshot=24**		**Snapshot=16**		**Snapshot=10**	
	ME	*WC*	*ME*	*WC*	*ME*	*WC*
100%	3.79	28.04	8.96	57.31	10.88	58.40
95%	2.34	15.48	5.79	31.23	7.99	31.20
90%	1.72	9.30	4.54	22.51	6.78	24.85

Finally we evaluate the impact of removing the small OD pairs on the accuracy of estimation for the remaining top OD flows. When flows are "removed" (i.e., set to zero) for the purpose of our method, the traffic they carry still appears inside the SNMP link counts. Thus, if we assume some OD flow has zero rate, then its actual rate will be transfered to some other OD pair retained in the estimation process. This will increase the inaccuracy in the flows being estimated.

We provide summary statistics for our errors as a function of the number of snapshots in Table II. Clearly both the average relative error and the worst case error increase as the number of snapshots is reduced. For example, the average error of 90% of our heavy OD pairs goes from 2% to 5% to 7% as the number of snapshots drops from 24 to 16 to 10 (respectively). Because the errors are so low with the full rank all OD flows system, even when few snapshots are used, the increased errors are still within an acceptable range (i.e., below the 10% target). We believe that this method that reduces the number of snapshots yet maintains acceptable error rates, has the potential to render the route change approach practical.

VIII. CONCLUSION

In this paper we propose a new approach for dealing with the ill-posed nature of traffic matrix estimation. We propose three solution enhancers: an algorithm for changing link weights so that the underlying linear system can be make full rank; a cyclo-stationary model to capture both long-term and short-term traffic variability, and a method for estimating the variance of OD flows. We show how these three elements can be combined into a comprehensive traffic matrix estimation procedure that dramatically reduces the errors compared to existing methods.

We demonstrated that our variance estimates can be used to identify the elephant OD flows, and we thus proposed a variant of our algorithm that addresses the problem of estimating the heavy flows in a traffic matrix. This is the first traffic matrix paper to propose a solution that focuses only on elephant OD flows. One of our key findings is that by focusing only on heavy flows, we can simplify the measurement and estimation procedure so as to render it more practical. This approach is more practical because it requires fewer routing changes and less

SNMP data collection (fewer measurement intervals). There is a small price to way in terms of accuracy of estimation in this case. However, we find that increasing the rank is so helpful that we can nevertheless keep the average errors consistently below the 10% carrier target error rate. Sometimes our errors reached as low as 1 or 2% (depending upon the scenario evaluated). We believe that this is the first proposed method to achieve average error rates in this range.

One of the limitations of our work is that it takes a long time to get a high confidence estimate of the covariance of the traffic matrix. In our future work, we hope to explore ways to reduce this measurement time.

APPENDIX

A. Viable Routing Changes Heuristic: VRC-Heuristic

The objective of the VRC-Heuristic is to identify the **minimal** set of snapshots needed so as to render the A matrix full rank. The suggested changes must not degrade the network performance customers experience. Thus, the proposed changes should cause neither excessive link loads nor lead to end-to-end delays that are larger than those specified in customer contracts. We thus allow carriers to input a *delay-limit* and a *load-limit* into our algorithm to ensure these constraints are met. Also operators are likely to prefer changing one or two link weights at the same time rather than changing a large numbers of links simultaneously, to avoid pushing the network "too far" from its operational working point. We limit the number of simultaneous weight changes to be three giving higher priority to single link weight changes. After identifying many candidate snapshots, our heuristic algorithm can be viewed as a cascade of pruning steps to extract those snapshots that achieve the goal while meeting the constraints.

Step 1: Pruning by Route Mapping. In this step we identify a set of snapshots, where each snapshot corresponds to an IGP weight change on a single link. Let $\Delta_h = \{\delta wh(p_1), \ldots, \delta w_h(p_P)\}$ be the set of potential IGP weight perturbations to apply to a given link l_h such to force OD pair p_k to flow on it. Recall that P is the number of OD pairs in the network. For each link $l_h \in \mathcal{L}$, we determine the set Δ_h, as follows. First, for an OD pair p, let $\theta_0(p)$ be the path length of the minimum cost path from the origin node of p to the destination node of p under the initial set of link weights $\mathcal{W}$. Let $\theta_h(p)$ be the length of the shortest path which is constrained to use link l_h. For each l_h and p, we define $\delta w_h(p) = \theta_0(p) - \theta_h(p)$ if $\theta_0(p) - \theta_h(p) < 0$, $\delta w_h(p) = -1$ if $\theta_0(p) - \theta_h(p) = 0$ and link l_h lies along exactly one shortest path for OD pair p, and $\delta w_h(p) = 0$ otherwise. We point out that any link weight change must result in a new weight that ties in the range $[W_{min}, W_{max}]$, corresponding to the range of weights allowed by the operator. Then, we prune the set of link weight perturbations for link l_h to the set Δ_h, where $\Delta_h = \{\delta w_h(p) : \delta w_h(p) < 0 \text{ and } w_h + \delta w_h(p) \in [W_{min}, W_{max}], \forall p \in \mathcal{P}\}$.

After considering all possible links l_h, we obtain a set of possible snapshots $\Delta = \cup_{h=1}^{L} = \Delta_h$, each of which corresponds to a single weight change and results in traffic from an OD pair being moved on at least one link.

Step 2: Pruning by Performance Constraints We now evaluate the set of candidate perturbations to see if they meet the *delay-limit* and *load-limit* constraints. We examine all the entries in the set Δ and build the set of candidate snapshots as a set of tuples $S = \{(l_h, W_{hi})\}$ where each tuple represents a deviation from the original weights set by setting the weight of the link l_h to the value w_{hi}, defined as $w_{hi} = w_h + \Delta w_h(p_i)$. For each tuple, we compute the associated routing matrix A_{hi}. For each candidate snapshot, we compute the new average end-to-end delay for all OD pairs and link-load by using the old TM available. If this violates the *delay-limit* or the *load-limit*, then we set aside the snapshot. For those snapshots that are within both the delay and load limits, we include them only if the corresponding routing matrix A_{hi} is not identical to that of a routing matrix for another included snapshot.

Step 3: Ordering the candidate snapshots. Now that we have a set of candidate snapshots, we need to determine which of those snapshots to include to generate our aggregate routing matrix. Conceptually, we could consider all possible subsets of the candidate snapshots, and compute the rank of the associated aggregate routing matrix for each subset of snapshots. Since the number of subsets is very large, this is computationally prohibitive. Instead, in this step we construct an ordering of the candidate snapshots according to a ranking function which reflects the the number of new OD pairs that become known exactly as a result of the candidate snapshot. The intuition is to pick snapshots that reveal many OD pairs exactly within a single snapshot.

We introduce some new notation used to define our ranking function. Let Δ_{hi} be the set of OD pairs whose routing is affected by changing the link weight on link l_h to the new value $w_{hi} = w_h + \Delta w_h(p_i)$. Let $\beta_{hi}(p)$ be a binary variable that is equal to 1 if the OD pair p uses the link l_h and 0 otherwise, with respect to the new set of link weights after changing the weight of link l_h to w_{hi} as described above. Let $R_{hi}(p_t)$ be the set of new links along the path used by OD pair p_t after applying the IGP weight w_{hi}, i.e., the set of links that are contained in a shortest path for p_t after changing the link weight w_h to w_{hi} but are not included in any shortest path for p_t for the original set of weights $\mathcal{W}$. We define the *ambiguity* of p_t after the weight change (l_h, w_{hi}) to be

$$m_{hi}(p_t) = min_{l_h \in R_{hi}} \sum_{p \in \Gamma_{hi}/p_t} \beta_{hi}(p) \quad \forall w_{hi} : (l_h, w_{hi}) \in S \tag{22}$$

which takes the minimum of these per-link sums across all the new links in the new path of OD pair p_t. The quantity $m_{hi}(p_t)$ is defined for a single OD pair p_t. Our ranking metricfor changing link l_h to the value w_{hi}, is defined over all ODpairs as follows.

$$M_{hi} = \sum_{p_i \in \Gamma_{hi}} (1/(Bm_{hi}(p_t) + 1)) \quad \forall w_{hi} : (l_h, w_{hi}) \in S \tag{23}$$

where B is a large parameter satisfying the constraint $P/(B+1) < 1$.

In defining the ranking function we have assumed that is better to know something exactly than simply to get new link counts without any complete dissggregation. The snapshots are then ranked according to the corresponding value of M_{hi} for each snapshot, where the snapshot with the largest value of M_{hi} is at the top of the list.

Step 4: Evaluation of candidate single link snapshots. We now evaluate the candidate snapshots in the order defined in Step 3. Each entry $(l_h, w_{hi}) \in S$ is evaluated sequentially by appending its associated routing matrix A_{hi} to the improved aggregate routing matrix A and computing the rank of the combined routing matrix. Only if the rank of the new A increases then A_{hi} is kept in A; otherwise the snapshot is discarded. The algorithm stops when the rank of the A matrix is equal to P.

Step 5: Multiple IGP weight changes and relaxation of requirements. For large networks performing only single weight changes might not be enough to guarantee that we have obtain a full rank routing matrix A. If this is the case, then we apply snapshots corresponding to weight changes on pairs of adjacent links and repeat the above steps starting from Step 1. In this context, a pair of links is defined to be adjacent if they are incident to a common node. The process continues until the aggregate routing matrix has full rank. If we exhaust all snapshots corresponding to a pair of link weight changes then we try snapshots corresponding to triplets of links. We consider only triplets of links that form "triangles", i.e., triplets of links of the form $\{(a,b),(b,c),(c,a)\}$. Moreover, for each triplet of links we only consider two possible weight changes, i.e., changing the weights of all three links to either W_{min} or all to W_{max}. The process continues until the aggregate routing matrix has full rank. If we reach this point and the rank of A is less than P, then we relax the delay and load contraints.

REFERENCES

[1] Y. Vardi, "Estimating source-destination traffic intensities from link data," *J. Amer. Statist. Assoc.*, vol. 91, no. 433, Mar. 1996.

[2] C. Tebaldi and M. West, "Bayesian inference of network traffic using link count data," *J. Amer. Statist. Assoc.*, vol. 93, no. 442, Jun. 1998.

[3] J. Cao, D. Davis, S. V. Weil, and B. Yu, "Time-varying network tomography: Router link data," *J. Amer. Statist. Assoc.*, vol. 95, no. 452, 2000.

[4] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot, "Traffic matrix estimation: Existing techniques compared and new directions," in *Proc. ACM SIGCOMM*, Pitsburgh, PA, Aug. 2002.

[5] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg, "Fast accurate computation of large-scale IP traffic matrices from link loads," in *Proc. ACM Sigmetrics*, San Diego, CA, Jun. 2003.

[6] Y. Zhang, M. Roughan, C. Lund, and D. Donoho, "An information theoretic approach to traffic matrix estimation," in *Proc. ACM SIGCOMM*, Karlsruhe, Germany, Aug. 2003.

[7] K. Papagiannaki, N. Taft, and A. Lakhina, "A distributed approach to measure IP traffic matrices," in *Proc. ACM IMC*, Oct. 2004.

[8] G. Liang and B. Yu, "Pseudo likelihood estimation in nework tomography," in *Proc. IEEE INFOCOM*, San Francisco, CA, Mar. 2003.

[9] A. Nucci, R. Cruz, N. Taft, and C. Diot, "Design of IGP link weight changes for estimation of traffic matrices," in *Proc. IEEE INFOCOM*, Hong Kong, Mar. 2004.

[10] A. Soule, A. Nucci, R. Cruz, E. Leonardi, and N. Taft, "How to identify and estimate the largest traffic matrix elements in a dynamic environment," in *Proc. ACM Sigmetrics*, Jun. 2004.

[11] A. Feldmann, A. Greenberg, C. Lunc, N. Reingold, J. Rexford, and F. True, "Deriving traffic demands for operational IP networks: Methodology and experience," *IEEE/ACM Trans. Networking*, vol. 9, no. 3, pp. 265–279, Jun. 2001.

[12] H. Stark and J. W. Woods, *Probability, Random Processes, and Estimation Theory for Engineers*. Englewood Cliffs, NJ: Prentice-Hall.

[13] A. Nucci, B. Schroeder, S. Bhattacharyya, N. Taft, and C. Diot, "IGP link weight assignment for transient link failures," in *Proc. 18th Int. Teletraffic Congr.*, Berlin, Germany, Aug. 2003.

Augustin Soule received the Ph.D. degree from the University of Pierre et Marie Curie in 2006, and the M.Sc. degree in electronics engineering from ISEP, France, in 2001.

He is currently a Researcher with Thomsom Paris Research Laboratory, Paris, France. His research interests lie in large-scale network measurement, traffic modeling and anomaly detection.

Antonio Nucci (M'99–SM'05) received the Dr. Ing. degree in electronics engineering in 1998 and the Ph.D. degree in telecommunications engineering in 2002, both from the Politecnico di Torino, Turin, Italy.

From 2001 to 2005, he worked in Sprint's Advanced Technology Laboratories in the IP research group. He is currently Chief Technology Officer at Narus Inc., Mountain View, CA. His research interests include traffic measurement, characterization, analysis and modeling, security and network design.

Rene L. Cruz (F'03) received the B.S. and Ph.D. degrees in electrical engineering from the University of Illinois at Urbana-Champaign in 1980 and 1987, respectively, and the S.M.E.E. degree from the Massachusetts Institute of Technology, Cambridge, in 1982.

Since 1987, he has been on the faculty at the University of California, San Diego.

Emilio Leonardi (M'99) received the Dr.Ing degree in electronics engineering in 1991 and the Ph.D. degree in telecommunications engineering in 1995, both from Politecnico di Torino, Italy.

In 1995, he visited the UCLA Computer Science Department. In 1999, he joined the High Speed Networks Research Group at Bell Labs/Lucent. He is currently an Associate Professor at the Dipartimento di Elettronica of Politecnico di Torino. His research interests lie in performance evaluation, queueing theory, packet switching, wireless networks, and optical networks.

Nina Taft (M'94) received the B.S. degree from the University of Pennsylvania, Philadelphia, in 1985, and the M.S. and Ph.D. degrees from the University of California at Berkeley in 1990 and 1994, respectively.

She is a Senior Researcher at Intel Research Berkeley. From 1995 to 1999, she worked at SRI International, and from 1999 to 2003, she was a member of the IP Group at Sprint Advanced Technology Laboratories. Her interests lie in traffic characterization and modeling, performance evaluation, network design and enterprise network security.