Performance of Estimated Traffic Matrices in Traffic Engineering

Matthew Roughan Mikkel Thorup Yin Zhang
AT&T Labs— Research, Shannon Laboratory
180 Park Avenue, Florham Park, NJ 07932
(roughan,mthorup,yzhang)@research.att.com

ABSTRACT

We consider the performance of estimated traffic matrices in traffic engineering. More precisely, we first optimize the routing in an IP backbone to minimize congestion with the estimated traffic matrix. We then test the performance of the resulting routing on the real traffic matrix.

Categories and Subject Descriptors

C.2.3 [**Computer-Communications Network**]: Network Operations—*network management, network monitoring*

General Terms

Measurement, Performance

Keywords

Traffic Estimation, Traffic Engineering, SNMP, OSPF, MPLS

1. INTRODUCTION

Estimating an Internet traffic matrix has received considerable attention in recent years. A traffic matrix provides the volume of traffic between every pair of ingress and egress points over a given time interval. Such information is essential to a variety of operational tasks ranging from router/link failure analysis to capacity planning and traffic engineering, for instance by route optimization.

When direct flow level measurements are available, accurate traffic matrices can be derived. Unfortunately, direct measurements require additional measurement infrastructure and it can be prohibitively expensive to instrument the entire IP network to collect such data. Recently, progress has been made on traffic matrix estimation and several methods have been proposed that attempt to derive traffic matrices from the link load data, which can be easily obtained via the Simple Network Management Protocol (SNMP). We call such a technique an SNMP-based *traffic matrix estimator*. These algorithms have been validated against real (but partial) traffic matrices (obtained through direct measurements) using common metrics such as the root mean squared error computed over all source-destination pairs. The resulting estimate contain errors of varying magnitude depending on the algorithm applied. It is not directly clear what impact these errors have on operational tasks, as different tasks may have quite different tolerance to the types and magnitude of the errors. For example, if all errors were concentrated on a single critical link, this could have a big impact on performance, yet a negligible impact in most standard error metrics. Thus we formulate the following general research question:

> *If we do traffic engineering based on the estimated traffic matrix, how well do we perform on the real traffic matrix?*

SIGMETRICS'03, June 10–14, 2003, San Diego, California, USA.
ACM 1-58113-66401/03/0006.

The concrete traffic engineering task we focus on in this paper is that of optimizing the routing so as to minimize max-utilization. Here the *utilization* of a link is the ratio of its load over its capacity, and the *max-utilization* is the maximum utilization over all links in the network. We call a technique for this task a *route optimizer*. First we will get an estimated traffic matrix $\hat{M}$ from a traffic matrix estimator. Then we apply a route optimizer to $\hat{M}$. The output routing $\hat{R}$ determines for each source and destination what fraction of the traffic should go on different paths from the source to the destination. Finally, we test the performance of the result $\hat{R}(M)$ of applying $\hat{R}$ to the true traffic matrix M.

2. TRAFFIC MATRIX ESTIMATORS

This section briefly describes three methods considered for estimating traffic matrices from link load data. The first two methods are based on so called "gravity models" while the third combines "gravity models" with "network tomography" methods. More details on the methods can be found in [2, 1].

Gravity models are often used by social scientists to model the movement of people, goods or information between geographic areas. Recently, variations on gravity models have also been proposed for computing traffic matrices.

At the heart of the gravity model approach is a proportionality assumption: the amount of traffic from a given source to a given sink is proportional to the total traffic to the output sink, independent of source. For example, in a gravity model for car traffic between cities the relative strength of the interaction between two cities might be modeled as proportional to the product of the populations divided by a distance related "friction" term. Similarly, the simplest possible gravity models for the Internet assume that the traffic exchanged between locations is proportional to the volumes entering and exiting at those locations, though in this case we assume the distance related term is a constant because interactions in the Internet are less distance sensitive. We refer to this as the *simple gravity model*.

It is possible to generalize the simple gravity model in a number of ways to take into account additional information provided by detailed link classification and routing policies. These papers [2, 1] have shown these gravity models to be significantly more accurate than the simple gravity models. We test the *generalized gravity model* of [1] in which additional information on points of ingress and egress for traffic flows can be incorporated to explicitly model hot-potato routing for traffic exchanged with peer networks.

By appropriate normalization, the gravity model solution is guaranteed to be consistent with the measured link loads at the network edge, but not necessarily so in the interior links. Alternatively, network tomography methods explicitly include the information measured from internal links. This information can be written as a set of linear constraint equations

$$\mathbf{x} = \mathbf{A}\mathbf{t}, \tag{1}$$

where $\mathbf{x}$ is a vector of the link measurements, $\mathbf{t}$ is the traffic matrix written as a column vector, and $\mathbf{A}$ is the routing matrix (whose terms give the fraction of traffic from a particular origin/destination pair that traverses each link).

In practice this set of equations is ill-posed, and so tomographic techniques from other fields have been used to deal with this difficulty. For a detailed description and comparison (using simple metrics) of a number of these methods see [2]. We shall consider a single such algorithm, *tomogravity*, [1] which displays good properties in terms of scaling, estimation accuracy, speed of computation, and robustness to errors. The method uses the generalized gravity model above as a prior (a kicking off point) and refines it using tomographic techniques to select an estimate of the traffic matrix $\hat{\mathbf{t}}$, that satisfies the constraint equations, and is closest to the gravity model according to some distance metric.

3. ROUTE OPTIMIZERS

General routing. In the most general form of routing, traffic from a source to a destination may be split arbitrarily over all possible paths between the source and destination. Finding a general routing minimizing max-utilization is an instance of the classical multicommodity flow problem. As described by Mitra and Ramakrishnan [3], the general routing solution can be implemented with the quite recent Multi-Protocol Label Switching (MPLS) protocol. Essentially, each path used is implemented as a label-switched path that the source uses for a certain fraction of its traffic to the destination. We call the technique that finds an optimal MPLS solution for a given traffic matrix an *MPLS optimizer*.

Traditional shortest path routing. The most commonly used intra-domain Internet routing protocols today are the the *shortest path* Interior Gateway Protocols (IGP): Open Shortest Path First (OSPF) and Intermediate System-Intermediate System (IS-IS). In these protocols, which are functionally the same, each link is associated with a positive weight, and the length of a path is defined as the sum of the weights of all the links on that path. Traffic is routed along the shortest paths. In cases of ties where several outgoing links are on shortest paths to the destination, the flow is split roughly evenly.

By default, Cisco routers set the weight of a link to be inversely proportional to its capacity – we refer to this setting as the *InvCap* weight setting. The weights of the links, and thereby the shortest path routes, can be changed by the network operators to optimize network performance.

Over the years, many methods have been presented that compute a set of link weights that minimize congestion in the resulting shortest path routing of a given traffic matrix (see references in [4]). We shall refer to such a method as an *OSPF optimizer*, though the results could equally be applied to IS-IS routing. We use the approach described in [4], which is based on so-called local search techniques. The method uses heuristics to iteratively try to improve the weight setting. The problem of finding an optimal weight setting is NP-hard [4], and so we cannot guarantee finding the true optimum. The quality of the final weight setting is affected by random choices made through the iterations, giving some variance in the quality of the outcome. For example, it is possible that we, by chance, get a better weight setting for the true traffic matrix from the estimated traffic matrix than we would get from the real traffic matrix itself, but the results below show that this random variation has little impact on real problems.

4. TESTS AND RESULTS

We tested our estimators and optimizers based on a day of hourly data from a AT&T's IP backbone running OSPF. That is, for each hour $i = 1, ..., 24$, we optimized the weight setting for the estimated traffic matrix from hour i and tested it on the true traffic matrix from hour i. Note that our "true" traffic matrices were really partial matrices derived from limited flow level measurements – the best we can get with our current measurement infrastructure. We then simulated the link loads that this true traffic matrix would generate and made our estimates from these link loads.

For tomogravity, the average relative error was 0.13, for general gravity it was 0.30, and for simple gravity was is 0.67. The max-utilization results are presented in Table 1.

route optimizer	traffic matrix	Max-utilization (%) Average	Max-utilization (%) Maximum
InvCap	Not relevant	79.9	100.0
OSPF	simple gravity model	57.5	67.2
OSPF	general gravity model	58.6	68.1
OSPF	tomogravity	47.1	57.7
OSPF	true	44.4	54.1
MPLS	tomogravity	53.5	68.8
MPLS	true	42.5	51.8

Table 1: The average and maximum max-utilization over the 24 hours. For proprietary reasons all numbers are scaled with the same secret factor.

The first interesting observation is that when applying OSPF to the estimated traffic matrices, the performance of general gravity is slightly worse than that of simple gravity. This sharply contrasts that general gravity is twice as good from the perspective of relative error. On the other hand, tomogravity is the best both with respect to relative error and with respect to max-utilization with OSPF. In fact, the OSPF max-utilization with tomogravity is only about 6% from that of the direct OSPF optimization over the true traffic matrix.

The second interesting contrast is OSPF versus MPLS optimization with respect to tomogravity versus true traffic matrix. MPLS is best possible when applied to the true traffic matrix. However, when presented the tomogravity estimates, it creates a significantly worse routing for the true traffic matrix than does OSPF. Thus our sub-optimal OSPF optimizer is much more robust to estimation errors. We note that this is not a direct criticism of MPLS as such, because MPLS is highly flexible in the method used to choose its routing. For instance, it seems to be a reasonably common practice to use an IGP to choose MPLS routes, and thus the above OSPF optimization would apply.

The last observation is that our OSPF optimization based on tomogravity gets within 11-12% of the optimal MPLS solution for the true traffic matrix, and which is the best possible among all routing protocols. Also, the OSPF tomogravity solution is about 50% better than the OSPF default InvCap. Thus we conclude that using existing techniques for generating traffic matrix from readily available SNMP data and for optimizing OSPF routing for a given traffic matrix, we obtained good routing for the real traffic. The method might be implemented with OSPF, IS-IS or MPLS, making it broadly applicable to todays IP networks.

5. REFERENCES

[1] 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*, June 2003.

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

[3] D. Mitra and K.G.Ramakrishnan, "A case study of multiservice, multipriority traffic engineering design for data networks," in *Proc. IEEE GLOBECOM*, pp. 1077–1083, 1999.

[4] B. Fortz and M. Thorup, "Internet traffic engineering by optimizing OSPF weights," in *Proc. IEEE INFOCOM*, pp. 519–528, 2000.