Optimizing the measurement of the traffic on large Networks: An Experimental Design Approach

Guillaume Sagnol*, Mustapha Bouhtou*, Stéphane Gaubert*

*INRIA Saclay / CMAP, Ecole Polytechnique
Route de Saclay, 91128 Palaiseau Cedex, France

*Orange Labs R&D, 38 rue du Général-Leclerc
92794 Issy-les-Moulineaux Cedex 9, France

Abstract

We study the classical problem of inferring the traffic on each Origin-Destination (OD) pair of a large IP network. The most recent methods take advantage of network-monitoring tools such as Netflow (Cisco Systems), which supplement the link measurements by direct information on the OD flows. The aim is to reduce the costs of deployment of Netflow, and to optimize its use. We formulate a combinatorial optimization problem, whose objective is to find the "best" set of interfaces on which Netflow should be activated. Our approach relies on an experimental design model, in which we minimize the variance of an unbiased estimator of a linear combination of the flows. We show that this problem can be solved very efficiently by Second Order Cone Programming, and we present a method called "Successive Optimal gamma Designs" to optimize the deployment of Netflow. We give experimental results comparing our method with previously proposed ones.

Keywords: *Traffic measurement, Experimental design, Combinatorial optimization, Second Order Cone Programming.*

1 Introduction

The problem of estimating Origin-Destination (OD) traffic matrices for backbone networks has recently attracted much interest from both Internet providers and the network research community [5, 9, 8, 14], because these traffic matrices serve as important inputs of a variety of network traffic engineering tasks. This estimation problem is generally stated as follows.

We are given the graph of the network, with its set of l edges (or links). Direct measurements are provided by the Simple Network Management Protocol (SNMP), which gives some statistics on the links (for instance, the number of bytes seen on each link in a 5 minutes window). We will denote these SNMP link counts by $Y^{\mathrm{SNMP}} = (y_1, \ldots, y_l)^T$. We are also given the set of routes among the network, that is to say a set of m OD pairs, and for each pair, the set of links that a byte need traverse to go from origin O to destination D. The information about the routing is classically gathered in the $l \times m$ incidence matrix A_0: this is a $0/1-$matrix whose (e,r)-entry takes the value 1 if and only if the OD pair r traverses edge e. More generally, the Internet provider routing policies may lead us to consider matrices in which $A_{0(e,r)}$ is a real number representing the fraction of the traffic from OD pair r that traverses link e.

The unknown in our problem is the vector of OD flows $X = (x_1, \ldots, x_m)$, where x_r is the number of bytes which have been traveling through OD pair r during the observation period. The following relation is easily seen to hold: $Y^{\mathrm{SNMP}} = A_0 X$.

In typical networks, we have $l << m$, and so, the estimation of the flow distribution X is a highly ill-posed problem. In particular, the previous system cannot have a unique solution X, and we need some additional constraints or information to ensure the identifiability of the model.

A way to introduce new constraints is to use a network-monitoring tool such as Netflow (Cisco systems). This was considered by the authors of [8], who proposed a scheme for selecting dynamically the flows to be measured by Netflow, in order to improve the accuracy of the traffic estimation. Of course, activating Netflow everywhere on the network yields an extensive knowledge of the OD flows. According to [6] however, activating Netflow on an interface of a router causes its CPU load to increase by 10 to 50%. Moreover, the deployment of Netflow on a network with heterogeneous routers may rise maintenance issues. It is thus of great interest to optimize the use of this tool.

A similar problem arises in the field of road traffic ; in that case, the pneumatic cables counting the number of vehicles driving on a road replace the SNMP data, and one can make surveys instead of using Netflow. The issue is thus to minimize the economic costs of the surveys needed to estimate the traffic between each origin and destination with a prescribed accuracy.

Related Work

The placement of Netflow has attracted much interest from the network research community [1, 2, 3, 4, 11, 13]. Most recent work includes Cantieni, Iannaccone, Barakat, Diot and Thiran [4], who interested themselves in the optimal rates at which Netflow should be sampled on each router, and formulated this problem as a convex minimization problem, which they solved using a projected gradient algorithm.

Song, Qiu and Zhang [11] used special criteria from the theory experimental design to choose a subset of measurements that Netflow should perform, and developed an efficient greedy algorithm to find a near optimal solution to this combinatorial problem.

In a previous work [2], we formulated the placement of Netflow as a combinatorial optimization problem, and showed that the Greedy algorithm always find a solution within $1 - 1/e \simeq 62\%$ of the optimal.

The main contribution of this paper is to give an alternative to the greedy approach of [11]: the Successive Optimal gamma-Designs approach (SOGD) introduced in Section 3 is not memory-consuming, it reduces to solving a sequence of moderate size second order cone programming problems which can be done rapidly by interior point methods, it may be tuned by the operator in order to give more weight to the most important flows. It yields results of a quality comparable to the greedy approach, whereas it is applicable to much larger instances.

The rest of this paper is organized as follows: The problem and the experimental design background are presented in Section 2. Next, we discuss previous methods to solve this problem, in particular the "Netquest" greedy approach proposed in [11]. The Successive Optimal gamma-Designs approach (SOGD), which is the main contribution of this manuscript, is presented in Section 3. Finally, we give some experimental results in Section 4.

2 Experimental design background and Problem statement

When Netflow is activated on an interface of the network, it will analyze the headers of the packets traversing this interface, and as a result we will have access to some statistics, such as the source and destination IP addresses, and the source and destination AS numbers of these packets. However, we are not directly interested in this information, because we are not trying to estimate the global run of the packets, but only the part of their run which is inside the network of interest, like the backbone of an autonomous system (AS).

Practically, and without any loss of generality [2], we will assume throughout this paper that when Netflow performs a measure on the k^{th} interface, we get a multidimensional measure Y_k which is a linear combination of the OD flows traversing interface k:

$$Y_k = A_k X \ . \tag{1}$$

We define the *design* variable w as the 0/1 vector of size s, where w_k equals 1 if and only if Netflow is activated on the interface k. The measurement vector Y is now the concatenation of the SNMP data Y^{SNMP} with all the Netflow measurements $(Y_k)_{\{k|w_k=1\}}$. The measurements are never exact in practice, and we have to deal with a noise ϵ, which is a result, among other things, of Netflow sampling and lost packets. This can be modeled as follows:

$$Y = A(w)\ X + \epsilon, \tag{2}$$

where $Y = [Y^T_{\text{SNMP}}, Y^T_{k_1}, ..., Y^T_{k_n}]^T$ and $A(w) = [A^T_0, A^T_{k_1}, ..., A^T_{k_n}]^T$.

Under the classical assumptions that we have enough measurements, so that $A(w)$ is of full rank, and that the noise has unit variance ($\mathbb{E}(\epsilon\epsilon^T) = I$), the best linear unbiased estimator is given by a Moore-Penrose generalized inverse : $\hat{X} = A(w)^\dagger Y$ (Gauss Markov Theorem), and its variance is:

$$\text{Var}(\hat{X}) = (A(w)^T A(w))^{-1}. \tag{3}$$

If we further assume that the noise follows a normal distribution $\mathcal{N}(0, I)$, then this variance is nothing but the inverse of the Fisher information matrix of the OD flows. We denote it by $M_F(w)$:

$$M_F(w) = A(w)^T A(w) = A_0^T A_0 + \sum_{k=1}^{s} w_k A_k^T A_k \ .$$

The experimental design approach consists in choosing the design w which maximizes a scalar function of the information matrix $M_F(w)$, which is concave and nondecreasing with respect to the natural ordering of the symmetric cone of positive semidefinite matrices, namely the Loewner ordering. For a more detailed description of the information functions, the reader is referred to the book of Pukelsheim [10], who proposes to use the matrix spectral functions Φ_p, which are essentially the "L_p-norms" of the vector of eigenvalues of the Fisher information matrix, but for $p \in [-\infty, 1]$. The function Φ_p is given by

$$\Phi_p(M) = \begin{cases} 0 & \text{for } p \leq 0 \text{ and M singular;} \\ \lambda_{\min}(M) & \text{for } p = -\infty \ ; \\ (\det(M))^{\frac{1}{m}} & \text{for } p = 0 \ ; \\ (\frac{1}{m} \text{ trace } M^p)^{\frac{1}{p}} & \text{otherwise,} \end{cases}$$

where we have used the extended definition of powers of matrices M^p for $p \in \mathbb{R}$: trace $M^p = \sum_{j=1}^m \lambda_j^p$.

We now give a mathematical formulation to the problem of optimizing the use of Netflow. Assume that the cost of deployment/activation of Netflow on interface k is c_k. If an Internet provider has a limited budget B, the *Netflow Optimal Deployment* problem is:

$$\begin{aligned} \max_{w\ \in\{0,1\}^s} \ & \Phi_p\Big(M_F(w)\Big) \\ \text{s.t.} \quad & \sum_k w_k c_k \leq B \end{aligned} \tag{4}$$

When the cost of deployment c_k is the same everywhere, the constraint is equivalent to deploy Netflow on no more than n interfaces. We call this special case the *unit-cost* case. For particular values of p, it is interesting to notice that we recover some classical optimal designs problems, known in the litterature as $A-$optimality ($p = -1$), $E-$optimality ($p = -\infty$), $D-$optimality ($p = 0$) and $T-$optimality ($p = 0$). For small positive values of p, we also recover the maximal rank design [2].

3 Resolution of the problem

In this section, we review previous methods to solve the *Netflow optimal deployment* problem, and we develop a new one. We limit our attention to methods that can be used on a large scale Network, say with more than 5000 OD pairs, and 100 interfaces.

Greedy Algorithm
The hardness and approximability results given presented in [2] suggest to use a greedy algorithm. On a network with $m = 5000$ OD pairs however, the computation of the objective function $\Phi_p(w)$ requires about 2 minutes on a PC at 4GHz, since it includes the diagonalization of a $m \times m$ matrix. Consequently, selecting only one among one hundred interfaces already requires more than 3 hours. In order to overcome this problem, the authors of [11] proposed to use the special values $p = 0$ or $p = -1$, which make the increment of the criterion efficiently computable thanks to Sherman-Morrisson like formulae. We adapted their idea to the case of block observations (1):

$$\left(M + A_k^T A_k\right)^{-1} = M^{-1} - M^{-1} A_k^T \left(I + A_k M^{-1} A_k^T\right)^{-1} A_k M^{-1} \ ,$$

$$\det(M + A_k^T A_k) = \det(M)\ \det\left(I + A_k M^{-1} A_k^T\right).$$

One could naturally object that at the beginning of the algorithm, when no interface has been selected yet, the initial observation matrix $M_0 = A_0^T A_0$ is not invertible. The authors of [11] remedy to this problem by regularizing the initial observation matrix: they set $M_0 = A_0^T A_0 + \varepsilon I$, with $\varepsilon = 0.001$. Although this trick may look arbitrary, it leads to very good results. However, the inconvenient of these greedy updates is that one needs to store the $m \times m$ matrix M^{-1} in memory, and to update it with the Sherman-Morrisson formula each time a new interface is selected by the greedy algorithm. The authors of [11] work on a similar experimental design problem, in which the observation matrix M has the nice property of being very sparse. Then, storing a sparse LU decomposition of the M is much more efficient than storing the full M^{-1} matrix, and it allows one to compute $M^{-1}A_k^T$. In our case though, M_0 is partially sparse only, and the LU decomposition is full. So this method may have reached its limit for our problem: it is unlikely that we may use it on a network with $m = 20000$ OD pairs without an additional effort to handle the storage of M^{-1}.

Convex Programming algorithms
A natural idea to solve the problem (4) is to replace the integer constraint $w \ \in \{0,1\}^s$ by

$$0 \leq w_k \leq 1, \forall k \in \{1, \ldots, s\}.$$

This relaxation lets problem (4) become a convex program. We can therefore apply different techniques to solve it, for instance projected gradient algorithms. Moreover, the solution of this relaxed program might be interpreted to some extent as a sampling of Netflow. The projected gradient algorithm for Problem (4) is described in [2], as well as rounding techniques to approximate the integer solution. The bottleneck of this algorithm is the spectral decomposition required to compute the gradient of Φ_p. For future work, we hope to implement second order techniques, by using appropriate approximations of the Hessian, whose direct computation is not tractable on large networks.

Successive Optimal gamma-Designs
The hardness of the *Optimal Netflow Deployment* is linked to the large dimension of the parameter that we want to estimate, which leads to large size covariance matrices. Rather than searching an estimator for the full parameter X, a natural idea is to estimate a linear combination $z = \gamma^T X$ of the flows, for which the best linear unbiased estimator and its variance are known [10] :

$$\hat{z} = \gamma^T A(w)^\dagger y \ , \qquad \mathrm{var}(\hat{z}) = \gamma^T M_F(w)^\dagger \gamma,$$

where $A^\dagger$ denotes the Moore-Penrose inverse of A. The variance of this estimator is a scalar, but it still depends (non-linearly) on a $m \times m$ matrix.
The following result shows that the minimization of $\mathrm{var}(\hat{z})$ is equivalent to a Second Order Cone Program (SOCP) ; the proof is omitted due to the lack of space, and will be published elsewhere. As in [7] (which handles the case in which the observation matrices A_i are vectors), the idea is to give a geometrical characterization of the $\gamma-$optimal designs, based on the intersection of the vectorial straight line directed by γ and the boundary of a special convex set.

Theorem 3.1. *Let w^* be the optimal (real) solution of the $\gamma-$optimal design problem:*

$$\min_{w\geq 0} \quad \gamma^T M_F(w)^\dagger \gamma$$
$$\sum_{i\geq 1} w_i c_i = B, \quad M_F(w) = \sum_{i=0}^{s} w_i A_i^T A_i.$$

And let $h^*, (\mu^*, z^*)$ be a pair of primal and dual solution of the SOCP:

$$(P-SOCP): \quad \max_h \quad \gamma^T h \qquad\qquad (D-SOCP): \quad \min_{\mu>0,z} \quad \sum_{i\geq 0} \mu_i$$
$$\|A_0 h\| \leq 1; \qquad\qquad A_0^T z_0 + \textstyle\sum_{i\geq 1} \sqrt{B/c_i} A_i^T z_i + \gamma = 0$$
$$\forall i \geq 1, \quad \|A_i h\| \leq \sqrt{\tfrac{c_i}{B}} \qquad\qquad \forall i \geq 0, \quad \|z_i\| \leq \mu_i.$$

Setting $T = \sum_{i\geq 1} \mu_i^*$, the following relations hold :

$$w_0^* = \mu_0^*/T, \quad \forall i \geq 1 \; w_i^* = \mu_i^* B/(c_i T),$$
$$T^2 = \gamma^T M_F(w^*)^\dagger \gamma \;\; .$$

This theorem shows how to compute the optimal weights w^* for the $\gamma-$combination of the flows by SOCP. This can be done very efficiently with interior points codes such as Sedumi [12]. Moreover, this method takes advantage of the sparsity of the matrices A_i, since SOCP codes are optimized to work with sparse matrices.

Another advantage of this method is the flexibility brought by the vector γ : The operator can choose this vector so that the weighted sum correlates the most important flows. In order to estimate all the flows, a possibility is to repeat the SOCP several times with vectors γ either chosen by the operator for their significance or tossed randomly on the unit sphere of dimension m, and finally to combine (for instance by taking the mean) the resulting optimal designs : we call this method SOGD (Successive Optimal gamma-Designs).

Once the optimal fractional w_i values are obtained, the deployment of Netflow can be done using a simple rounding heuristic: one can sort the interfaces according to the ratio $\frac{w_i}{c_i}$, and activate Netflow sequentially on these sorted interfaces until the budget constraint is violated.

We may ask ourselves whether it is a good idea to add the constraint $\mu_i B \leq c_i \sum_{k\geq 1} \mu_k$ in Problem $(D-SOCP)$, in order to ensure that $w_i \leq 1$. However, our proof does not seem to adapt to this case, and we can not guaranty that the w^* that one would obtain with this additional constraint is the solution of the $\gamma-$optimal design problem in $[0,1]$. Moreover, the optimal fractional w_i values never exceed 1 in practice thanks to the budget constraint, and even if it would, it does not prevent one from applying the rounding heuristic proposed above (although we loose the sampling interpretation of the vector w^*).

4 Experimental Results

We now present some comparisons between the methods presented above : experiments were carried out on real data from France Telecom Opentransit backbone (100 nodes, 267 links and 5591 OD-pairs). The inference was made on dynamic flows observed during 50 time steps, using the widespread methodology of entropic projections [9, 8], and the gravity model as initial estimate [9]. As our sample of Netflow data was incomplete, we simulated the true value of the unmeasured flows, and we generated both SNMP and Netflow measurement using relation (2).

Table 1 compares the approaches presented in the previous section in terms of both CPU time and the size of the smallest full-rank design found, i.e. the design with the smallest number of nodes where one should activate Netflow in order to infer exactly all the flows. The computation was made in the unit cost case, and we assumed that Netflow measured each flow traversing the interfaces where it is activated. The performance of the SOGD method is outstanding in term of CPU time, although the greedy algorithm

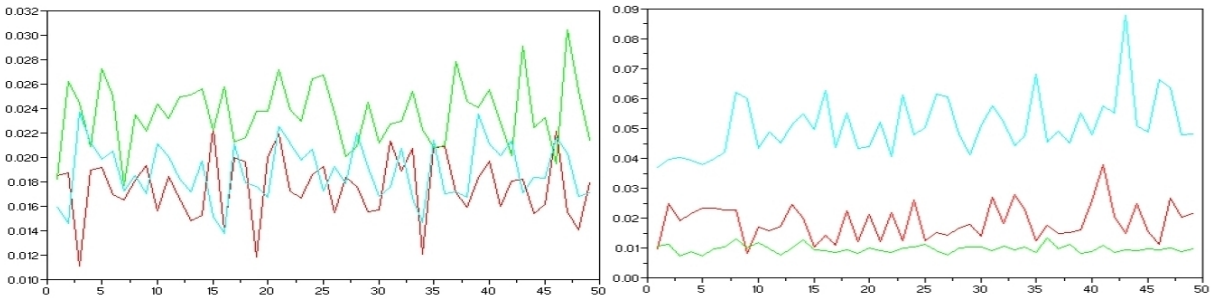

Figure 1: WAE vs time with Netflow on 10 nodes (left) and 17 nodes (right) selected by Greedy (red), SOGD (green), and P.Gradient (blue).

Algorithm	Smallest full-rank design	CPU
Greedy ($p = -1$)	35	3h
Projected gradient	39	65h
SOGD	36	4min
SOGD + Rounding heuristic	35	16min

Table 1: Comparison based on the smallest full-rank design found

finds a smaller full-rank design. Nevertheless, a simple rounding heuristic [2] allows one to obtain the same full-rank design that the one provided by the Greedy algorithm. Table 1 also shows the limit of direct convex programming methods, which fail to be efficient on large scale networks.

Figure 2 compares the deployment of Netflow on 10 nodes of the network obtained by the greedy algorithm (in red) and by SOGD (circled in blue). One can notice that these deployments are the same except for 2 nodes. The metric used to compare the designs is the weighted average error(WAE) : At time step t, it is defined as the mean of the vector of relative errors $|X_t - \hat{X}_t|/X_t$, where $\hat{X}_t$ is the vector found by entropic projection of $\hat{X}_{t-1}$ over the space$\{X \geq 0 | A(w)X = Y_t(w)\}$. Figure 1 shows the weighted average error on the traffic estimation for the 1000 largest flows, which represent 97% of the traffic due to their lognormal distribution [9]. If the estimation is better with the greedy deployment for Netflow activated on 10 nodes, the SOGD method yields better results for 17 nodes.

Another important comparison of the deployment of Netflow is the number of flows correctly estimated by each method. Figure 3 shows the distribution of the errors for the 1000 largest flows (averaged over time), when Netflow is activated on 17 nodes. It is very satisfactory to notice that more than 97% of the 1000 largest flows are estimated with an average error below 1% by the SOGD mthod. We hope to improve this result by tuning the vector γ in order to estimate the largest flows in priority.

Finally, Figure 4 shows how the weighted temporal error falls when we add Netflow measurements, with nodes selected by SOGD. This suggests that 17 interfaces are enough to infer correctly the traffic matrix. The additional effort to obtain a design of full-rank is both huge and of limited interest, since no improvement on the estimation is to notice between 17 and 35 nodes. The reason why the flows are not inferred exactly with a full rank matrix is the error ϵ on the measurements (2).

5 Conclusions

In this paper, we have proposed a new method to optimize the traffic measurement, based on the estimation of a sequence of linear combinations of the flows, rather than on the estimation of the full vector of flows. This method remains tractable for very large instances, and it allows one to identify the traffic accurately. Our computational results also suggest that it may be unnecessary to activate measurement tools such as Netflow with a design of full-rank, since we achieve results of the same quality by activating Netflow only on one half of the required nodes, the missing part of the traffic being recoveredmethod.

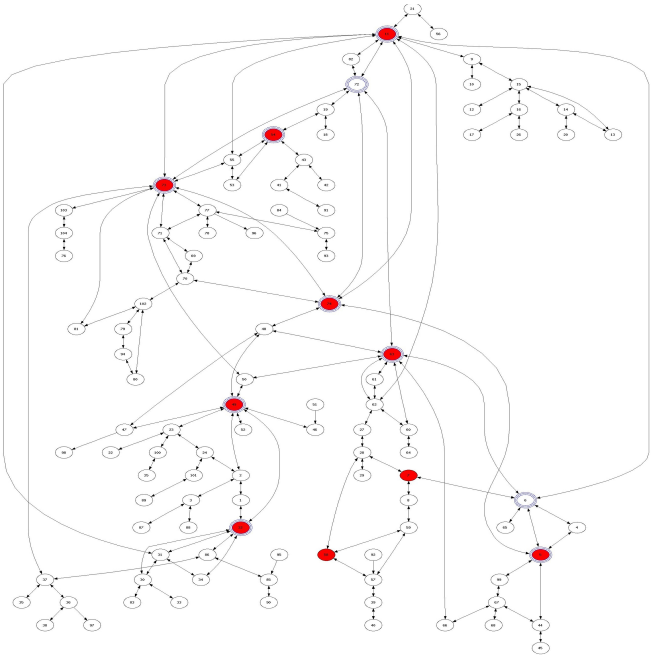

Figure 2: Optimal Deployment of Netflow on 10 nodes. Greedy in Red, SOGD circled in blue.

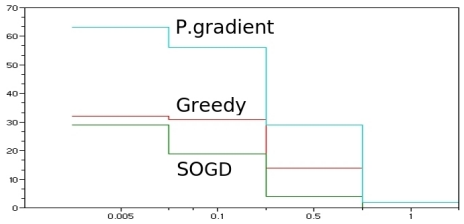

Figure 3: number of flows with an error of estimation larger than $0.5\%, 10\%, 50\%$ and 100%.

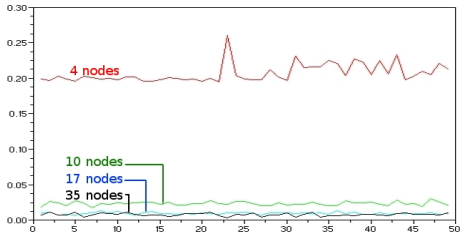

Figure 4: Evolution of the temporal error with the number of nodes with Netflow.

References

[1] P. Bermolen, S. Vaton, and I. Juva. Search for optimality in traffic matrix estimation : a rational approach by Cramer- Rao lower bounds. In *NGI'06*, pages 224 – 231, 2006.

[2] M. Bouhtou, S. Gaubert, and G. Sagnol. "optimization of network traffic measurement: a semidefinite programming approach". In *ENGOPT*, Rio De Janeiro, Brazil, 2008.

[3] M. Bouhtou and O. Klopfenstein. Robust optimization for selecting Netflow points of measurement in an ip network. In *GLOBECOM '07, IEEE.*, pages 2581–2585, 2007.

[4] G. Cantieni, G. Iannaccone, C. Barakat, Ch. Diot, and P. Thiran. Reformulating the monitor placement problem: Optimal network-wide sampling. In *CISS*, pages 1725–1731, 2006.

[5] J. Cao, D. Davis, S. Vander, and W.B. Yu. Time-varying network tomography : Router link data. *Journal of the American Statistical Association*, 95:1063–1075, 2000.

[6] CISCO. Netflow performance analysis, technical white paper, May 2007.

[7] G. Elfving. Optimum allocation in linear regression theory. *The Annals of Math. Stat.*, 23:255–262, 1952.

[8] G. Liang, N. Taft, and B. Yu. A fast lightweight approach to origin-destination ip traffic estimation using partial measurements. *IEEE/ACM Trans. Netw.*, 14(SI):2634–2648, 2006.

[9] A. Medina, N. Taft, S. Battacharya, C. Diot, and K. Salamatian. Traffic matrix estimation: Existing techniques compared and new directions. In *Proc. of SIGCOMM*, Pittsburgh, Aug. 2002.

[10] F. Pukelsheim. *Optimal Design of Experiments*. Wiley, 1993.

[11] H.H. Song, L. Qiu, and Y. Zhang. Netquest: A flexible framework for largescale network measurement. In *SIGMETRICS'06*, St Malo, France, 2006.

[12] J.F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999.

[13] H. Zang and A. Nucci. Optimal netflow deployment in IP networks. In *19th International Teletraffic Congress (ITC)*, Beijing, China, August 2005.

[14] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg. Fast accurate computation of large-scale ip traffic matrices from link loads. In *SIGMETRICS '03*, pages 206–217, 2003.