COPE: Traffic Engineering in Dynamic Networks

Hao Wang[⋆] Haiyong Xie[⋆] Lili Qiu[†]
Yang Richard Yang[⋆] Yin Zhang[†] Albert Greenberg[§]
AT&T Labs – Research[§] Univ. of Texas at Austin[†] Yale University[⋆]
{hao.wang,haiyong.xie,yang.r.yang}@yale.edu {lili,yzhang}@cs.utexas.edu
albert@research.att.com

ABSTRACT

Traffic engineering plays a critical role in determining the performance and reliability of a network. A major challenge in traffic engineering is how to cope with dynamic and unpredictable changes in traffic demand. In this paper, we propose COPE, a class of traffic engineering algorithms that optimize for the expected scenarios while providing a worst-case guarantee for unexpected scenarios. Using extensive evaluations based on real topologies and traffic traces, we show that COPE can achieve efficient resource utilization and avoid network congestion in a wide variety of scenarios.

Categories and Subject Descriptors: C.2.2 [Computer Communication Networks]: Network Protocols; C.2.3 [Computer Communication Networks]: Network Operations—*Network Management*

General Terms: Algorithms, Design, Management, Performance, Reliability.

Keywords: COPE, Traffic Engineering, Unpredictable Traffic, Optimization, Oblivious Routing.

1. INTRODUCTION

Traffic engineering (TE) has become an indispensable tool used by many autonomous systems (ASes) to select routes which effectively utilize their network resources. This is particularly important given the high cost of network assets and the highly competitive nature of the Internet ISP market [8, 9, 46]. The importance of traffic engineering has motivated many studies in the last few years, and quite a few traffic engineering algorithms were recently proposed (*e.g.*, [4, 6, 7, 19, 22, 23, 24, 25, 26, 27, 34, 36, 44, 48, 49, 50, 54]).

Traffic characteristics are a major factor affecting the design of traffic engineering algorithms. Unfortunately, for many ASes, although their traffic demand can be relatively stable most of the time, there exist time periods during which traffic can be highly dynamic, containing unpredictable traffic spikes that ramp up extremely quickly, leaving no time for a traffic engineering algorithm to re-compute or adjust. We recently examined the traffic traces of several backbone networks and found that there exist short time periods during which traffic demand can increase by at least one order of magnitude. Highly unpredictable traffic variations have also been observed and studied recently by other researchers (*e.g.*, [27, 37, 38, 41, 47, 54]). To further confirm the likelihood of observing highly unpredictable traffic spikes in real-life, we surveyed the operators of some large ASes and received reports of highly unpredictable traffic patterns in their daily operations. Many factors contribute to the highly unpredictable nature of Internet traffic: outbreaks of worms/viruses, outages or routing changes of major ISPs, the occurrence of natural disasters, denial-of-service attacks, and flash-crowd effects due to major news events. For many cases, traffic spikes occur exactly when the networking service is most valuable! In addition, with more bursty UDP-based multimedia traffic, more dynamic traffic such as that from overlay networks [30], and more networks adopting traffic engineering, variability in traffic could increase further.

It is important that traffic engineering handle sudden traffic spikes. If a traffic engineering algorithm is not prepared for them, it may cause network links and routers to be unnecessarily overloaded. Overloaded links and routers can cause long delay, high packet loss rates, reduced network throughput (*e.g.*, TCP flows), BGP session reset, and even router crashes. These reduce network reliability and efficiency, and may violate increasingly stringent service level agreements (SLAs), leading to potential financial penalties.

Despite the importance of handling traffic spikes, most of the proposed traffic engineering algorithms belong to a type of algorithms which we call *prediction-based TE*, and these algorithms optimize their routing without preparing for unpredictable traffic spikes. Such an algorithm first collects a set of sample traffic matrices, and then computes a routing to optimize the performance, based only on these samples. For instance, the algorithm can optimize for the average or the worst-case cost over these samples. An advantage of this type of algorithms is their potential performance gain. When the network traffic is relatively stable and the real traffic is similar to the samples based on which the routing is computed, these algorithms can achieve near-optimal performance. However, since these algorithms optimize routing specifically for these samples, when the real traffic deviates substantially from the samples (*e.g.*, during the presence of traffic spikes), the computed routing may perform poorly. An extreme case of prediction-based TE is online adaptation. An advantage of this scheme is that if it can converge quickly, it does not need to collect many samples or make prediction. However, when there are significant and fast traffic changes, such a scheme can experience a large transient penalty.

As motivation, we show using real traffic traces that when ISPs use prediction-based TE and unexpected traffic spikes occur, the traffic intensity of some links may well exceed their link capacities. For example, using the real Abilene topology and traffic traces, we show that for bottleneck links, the traffic intensity generated by all three prediction-based algorithms we evaluated exceeds link capac-

SIGCOMM'06, September 11–15, 2006, Pisa, Italy.

ity during traffic spikes, and some reaches 2.44 times link capacity, while for an optimal algorithm, no link receives traffic above 50% of its capacity. Such large performance penalties arise when actual traffic demands deviate significantly from prediction. Therefore, it is important that a traffic engineering algorithm is robust when such deviations occur.

One way to deal with unpredictable traffic spikes is oblivious routing (*e.g.*, [6, 7, 27, 42, 54]). In oblivious routing, a routing that is independent of the traffic matrix is computed, and thus has the potential to handle traffic spikes well. A potential drawback of oblivious routing, however, is its sub-optimal performance for normal traffic, which may account for the vast majority of time periods. For example, the optimal *oblivious ratio* of arbitrary symmetric networks can grow logarithmically as they scale up [7]. Recently, Applegate and Cohen [7] computed the oblivious ratio of several real network topologies. Although they discovered that the ratio is typically only around 2, they also commented that overhead at this level "is far from being negligible to working ISPs."

Besides rapid traffic fluctuations, interdomain routing poses another set of challenges to traffic engineering. First, interdomain routing introduces point-to-multipoint demand; that is, there can be multiple equally-good egress points for some external destinations in the BGP decision process [17]. Thus, it is up to the intradomain routing determined by traffic engineering to break the tie. Since egress links may become the bottlenecks of the network [16], this tie-breaking can affect the congestion of the network. Second, although interdomain routes for most traffic volumes can be stable [32, 51], there are BGP routing changes that can cause significant shifts of traffic [37]. In particular, with the dynamic nature of the global Internet, the available interdomain routes of an AS can fluctuate as its peers announce and withdraw interdomain routes, or even reset their eBGP sessions. Thus, the intradomain routing determined by traffic engineering should be robust against such interdomain route changes.

In this paper, we propose novel traffic engineering algorithms to handle both dynamic traffic and interdomain routing. Our key insight is that we can use an efficient and easily implementable technique to guarantee worst-case performance under all traffic demands. By choosing a worst-case guarantee that is just a small percentage above lowest possible, we can optimize routing for predicted demands, and significantly improve common-case performance.

Based on this insight, we design a new class of traffic engineering algorithms, called *Common-case Optimization with Penalty Envelope* (COPE). Our algorithms combine the best of oblivious routing and prediction-based optimal routing. The penalty bound component of COPE is inspired by the pioneering work of oblivious routing [7]. Thus, COPE can bound the worst-case performance penalty to ensure acceptable performance when the network experiences unpredictable changes. But unlike oblivious routing, COPE optimizes routing for predicted demands to achieve high efficiency under normal network conditions. Therefore COPE can achieve close-to-optimal performance in the common case in our real traffic traces, whereas oblivious routing can be 30% - 90% worse than optimal.

We extend COPE to deal with interdomain routing. To handle the point-to-multipoint nature of interdomain demands and the dynamics of interdomain routes, we compute routing that is robust to changes in interdomain routes and yet responsive to traffic patterns.

The rest of the paper is organized as follows. In Section 2, we overview the related work. In Section 3, we present common-case optimization with penalty envelope (COPE). In Section 4, we compare the performance of COPE with the state-of-art approaches using real traffic traces and network topologies. We further extend COPE to handle interdomain routing, and describe our preliminary results in Section 5. We conclude in Section 6.

2. RELATED WORK

There is a large body of literature on traffic engineering. In the interest of brevity, we review only the most related work.

Intradomain traffic engineering has received significant attention in the research community. Many interesting traffic engineering algorithms and mechanisms have been proposed. Due to the flexibility and increasing popularity of MPLS [46], many recent studies focus on MPLS-based traffic engineering. We broadly classify this work into the following two categories: (i) traffic engineering for predicted traffic demands, and (ii) oblivious routing.

The algorithms in the first category share the following features: they maintain a history of observed traffic demand matrices, and they optimize for the representative traffic demand matrices extracted from the observed traffic during a certain history window. For example, Sharad *et al.* [4] use a traffic matrix in a one-hour window during daily peaks as the representative demand. Zhang *et al.* [49, 50] consider multiple representative traffic matrices and find an optimal set of routes to minimize expected or worst-case cost for these representative matrices. Note that in their approach, the worst case is only among the samples, not all possible traffic matrices. In [52], Zhang and Ge try to identify critical matrices from past history, and then conduct traffic engineering based on these matrices. It might be possible to extend prediction-based optimization using robust optimization (*e.g.*, [14]), but it will be challenging to estimate the variation set of parameters. MATE [19] and TeXCP [26] conduct online traffic engineering and react to instantaneous traffic demands. An advantage of these dynamic algorithms is that if they can converge quickly, they do not need to collect many samples or make prediction. However, when there are significant and fast traffic changes, these algorithms can experience a large transient penalty, as we will show in Section 4.

The second category of algorithms is oblivious routing (*e.g.*, [6, 7, 10, 11, 27, 28, 31, 42, 54]). In oblivious routing, routes are computed to optimize the worst-case performance over all traffic demands. Therefore the computed routes are prepared for dynamic changes in traffic demands. In their pioneering work [7], Applegate and Cohen propose an efficient algorithm to compute the worst-case oblivious routing for real networks. They also extend oblivious routing to compute failure scenarios [6]. They found that the oblivious ratio is typically around a factor of 2. A penalty as high as 100% may be acceptable when traffic demands are completely unpredictable, but it is a high cost to pay under predictable demands. In other words, oblivious routing takes a pessimistic point of view and may not be appropriate in relatively stable periods or stable networks.

Our approach is inspired by both prediction-based routing and oblivious routing, and combines the best of both approaches. It optimizes routing for predicted demands to achieve high efficiency under normal network conditions; in the meantime it also bounds the worst-case performance penalty to ensure acceptable performance when the network experiences unpredictable changes.

There are also recent studies on the interaction of intradomain traffic engineering with interdomain routes and traffic. Examples include evaluation (*e.g.*, [3, 37, 39, 40, 45]) and design (*e.g.*, [16, 20, 21, 22]). Recently, researchers observed that intradomain traffic engineering within an AS can cause substantial traffic changes outside the AS (*e.g.*, [4, 15, 37]). For example, Agarwal *et al.* report in [4] that for an operational tier-1 ISP, intradomain traffic engineering can cause up to 25% of its traffic to a neighboring AS to shift the exit point. Such traffic changes could trigger routing changes at the neighboring AS, and result in network instability. Motivated by these studies, we further extend COPE to handle interdomain routing.

3. OPTIMAL TE WITH TOLERANCE

In this section, we focus on a single AS. We assume that the egress point of each external destination is known and fixed. We will extend to the case of interdomain routing in Section 5.

3.1 Background Definitions

One major objective of the traffic engineering of an AS is to determine routing so as to minimize congestion. For concreteness, in this paper we measure network congestion using metrics based on maximum link utilization (MLU), as it is a commonly used metric in many studies (*e.g.*, [4, 6, 7, 26]). Another possibility would be to use network cost [24, 25] to measure congestion. Our scheme is directly applicable to optimizing for this alternative metric. We will discuss generalization in Section 3.3.

An AS is represented by a graph $G = (V, E)$, where V is the set of routers, and E is the set of network links. The capacity of link (i, j) from node i to node j is denoted by $c(i, j)$. For intradomain routing, we assume that the graph is stable during the operation. When the network topology changes (*e.g.*, a link that carries substantial amount of traffic fails), the routing computed by traffic engineering is no longer valid and has to be updated. For important intradomain links, a good recovery strategy is to pre-compute routing for each failure scenario [6]. Our COPE algorithms can be extended to deal with such scenarios.

The input to traffic engineering is traffic demand matrices (TM). We represent a TM as a set of traffic demands $D = \{d_{ab} | a, b \in V\}$, where d_{ab} is the demand for the origin-destination (OD) pair $a \to b$.

The output of traffic engineering is routing. Since around half of the ISPs run MPLS in their core [26], and more ASes are starting to deploy MPLS, we focus our study on MPLS-based routing. Slightly different from MPLS-style path routing, we use link-based routing [7, 13]. A link-based routing f is specified by a set of values $f = \{f_{ab}(i, j) | a, b, i, j \in V\}$, where $f_{ab}(i, j)$ specifies the fraction of demand from a to b that is routed over the link (i, j). One can convert link-based routing to standard MPLS path-based routing [5, 29], to shortest-path implementable routing [44], and to OSPF equal weight-split routing [36]. As we show in Appendix, we can incorporate shortest-path implementation considerations by adding penalty terms into performance metrics. Unless otherwise stated, routing refers to link-based routing in this paper.

For f to be a routing, the values of $f_{ab}(i, j)$ for the OD pair $a \to b$ should specify a flow of value 1 from a to b. For an actual demand d_{ab} for the OD pair $a \to b$, the contribution of this demand to the flow on a link (i, j) is $d_{ab} f_{ab}(i, j)$. The constraints on the routing variables $\{f_{ab}(i, j)\}$ are flow conservation and non-negativity, which can be defined by the following equations:

$$\begin{cases} \forall a \neq b, \forall i \neq a, b: & \sum_{(i,j)\in E} f_{ab}(i, j) - \sum_{(j,i)\in E} f_{ab}(j, i) = 0; \\ \forall a \neq b: & \sum_{(a,j)\in E} f_{ab}(a, j) - \sum_{(j,a)\in E} f_{ab}(j, a) = 1; \\ \forall (i, j) \in E: & f_{ab}(i, j) \geq 0. \end{cases} \tag{1}$$

The maximum link utilization (MLU) of a routing f on a TM D is defined as the maximum of traffic to capacity ratios of all links:

$$U(f, D) = \max_{(i,j)\in E} \sum_{a,b} \frac{d_{ab} f_{ab}(i, j)}{c(i, j)}. \tag{2}$$

An *optimal* routing for a given TM D is a routing that minimizes the maximum link utilization. Formally, the *optimal utilization* for a TM D is given by

$$OU(D) = \min_{f \text{ is a routing}} U(f, D). \tag{3}$$

The *performance ratio* of a given routing f on a given TM D is defined as

$$P(f, D) = \frac{U(f, D)}{OU(D)}. \tag{4}$$

It measures how far the routing f is from being optimal on TM D. $P(f, D) = 1$ indicates that the routing f is optimal. A higher ratio indicates that the performance is farther away from the optimal.

To account for fluctuations in network traffic, we may consider multiple traffic demand matrices. Given a set of TMs $\mathcal{D}$, there are multiple ways to extend a performance metric defined on a single TM D to the set $\mathcal{D}$. Since our objective is on robustness, we consider the worst case extension of $U(f, D)$ and $P(f, D)$.

Extending $U(f, D)$, we define the *maximum MLU* of a routing f on the set $\mathcal{D}$ as

$$U(f, \mathcal{D}) = \max_{D\in\mathcal{D}} U(f, D). \tag{5}$$

We refer to a routing that minimizes the maximum MLU on $\mathcal{D}$ as an *MLU optimal routing* on $\mathcal{D}$, and the corresponding maximum MLU as the *optimal MLU* on $\mathcal{D}$.

Extending $P(f, D)$, we define the *maximum performance ratio* of a routing f on the set $\mathcal{D}$ as

$$P(f, \mathcal{D}) = \max_{D\in\mathcal{D}} P(f, D). \tag{6}$$

We refer to a routing that minimizes the maximum performance ratio on $\mathcal{D}$ as a *performance-ratio optimal routing* on $\mathcal{D}$, and the corresponding maximum performance ratio as the *optimal performance ratio* on $\mathcal{D}$. When $\mathcal{D}$ is the complete traffic demand space containing all non-negative traffic demands, the performance-ratio optimal routing is referred to as the *oblivious routing*, and the optimal performance ratio is referred to as the *oblivious ratio*.

3.2 Optimal TE with Convex-Hull Prediction

We start with a type of robust prediction-based TE algorithms. Assume that a traffic engineering system has collected a set of TMs $\{D_1, \ldots, D_H\}$ during some time interval, where H is the number of TMs collected. To compute the routing for the next interval, the TE system needs to predict the TM that may appear during the next interval. There can be many predictors. A large class of predictors (*e.g.*, exponential moving average) essentially estimate the TM of the next interval as a convex combination of the previously seen TMs. Aggregating the predictions of all such predictors, we obtain the convex hull of $\{D_1, \ldots, D_H\}$.

Let $\mathcal{D}$ be the convex hull of the set of TMs $\{D_1, \ldots, D_H\}$. More specifically, the convex hull can be constructed using convex combinations of the TMs in $\mathcal{D}$, namely, $\sum t_h D_h$, where t_h is a coefficient between 0 and 1, $\sum_h t_h = 1$, and D_h is the h-th traffic matrix. Then the problem of optimal TE with convex-hull prediction is to compute the MLU or performance-ratio optimal routing over the set $\mathcal{D}$.

One advantage of a convex-hull-based predictor is its monotonicity (*i.e.*, if the convex hull is continuously maintained, it will always grow). Specifically, the monotonicity property leads to the following stability result:

PROPOSITION 1. *If interdomain BGP stability condition (*e.g.*, [43]) is satisfied, then intradomain traffic engineering using convex hull eventually converges.*

3.3 Common-Case Optimization with Penalty Envelope (COPE)

The convex-hull-based TE is effective when future demands fall into the convex hull. However, traffic fluctuation may make future demands fall outside the convex hull. In this case, the performance may degrade significantly. One way to handle this issue is to expand the convex hull to include more traffic demands. We can expand the corresponding convex hull by letting the convex combination coefficients t_h take values less than 0 or larger than 1. Then we can optimize routing for all traffic demands that fall into the expanded convex hull. Such expansion could help us to tolerate changes in traffic demands to a certain extent. However, there is a significant trade-off between the degree of expansion and the performance optimality. In an extreme, the convex hull can be expanded to include all traffic demands, which results in oblivious routing. This is robust against arbitrary possible traffic changes, but does not provide the best performance for normal demands.

To address the problem, we separate the optimization for the common (predicted) cases and the bound on the worst cases. In particular, we propose a novel approach based on the notion of *penalty envelope*. It guarantees worst-case performance under arbitrary possible traffic demand while achieving close-to-optimal performance under predictable demands.

DEFINITION 1. *A routing f is said to have* MLU *(or* performance-ratio*) penalty envelope $\bar{r}$ if the maximum MLU (or performance ratio) of f on the whole set of possible traffic demands is no more than $\bar{r}$.*

A penalty envelope restricts the set of possible routing to those with maximum MLU or performance ratio less than or equal to $\bar{r}$. With the penalty envelope as a safeguard, a prediction-based TE algorithm can then search the optimal routing f for the predicted traffic demands, so long the routing satisfies the penalty envelope. We call such a scheme common-case optimization with penalty envelope (COPE).

The general COPE scheme can be defined as follows. Let $\mathcal{X}$ be the set of all possible TMs, and $\mathcal{D} \subset \mathcal{X}$ the set of predicted TMs. Let $o(f,x)$ be the objective function of applying routing f to TM x. Let $o(f,\mathcal{D})$ be the aggregated objective function on the set $\mathcal{D}$. The aggregation can be done, for example, by taking the maximum, or by taking some type of weighted average. Let $c(f,x)$ be the penalty function. Note that for both the objective and penalty functions, lower values are better. It can also be the case that $o(f,x) = c(f,x)$. Then the general setting is to find the routing f that minimizes $o(f,\mathcal{D})$, under the constraint that the penalty over the whole set $\mathcal{X}$ of possible traffic demands (and thus includes those in $\mathcal{X} - \mathcal{D}$) is bounded by a penalty envelope $\bar{r}$. Formally, the formulation is:

$$\begin{array}{rl} \min_f & o(f,\mathcal{D}) \\ \text{subject to} & f \text{ is a routing, } i.e., (1)\ ; \\ & \forall x \in \mathcal{X} : c(f,x) \le \bar{r}. \end{array}$$

Figure 1: General COPE framework.

Figure 2 illustrates the basic idea of the COPE scheme. This scheme gives us a novel, simple, yet effective tool to handle dynamic networks with mostly normal traffic but sometime unexpected (yet possible) traffic demands.

3.4 Implementing Penalty Envelope

There are two remaining issues in implementing penalty envelope. The first is how to choose the penalty envelope $\bar{r}$. The algorithm we use is the following. First, we compute $c^*(X) = min_{f,x\in X}\, c(f,x)$. Then we set $\bar{r} = \alpha c^*(X)$, where α is a scale-up factor. As we will show in Section 4, by choosing α slightly higher than 1, we can achieve performance close to optimal for most common cases.

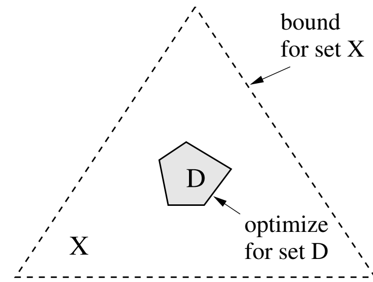


Figure 2: Illustration of COPE. The system objective is to choose the routing f which is optimal for the predicted set $\mathcal{D}$ under the constraint that its penalty in the set of all possible traffic demands $\mathcal{X}$ is bounded.

The second key challenge to the COPE framework is whether we can efficiently incorporate penalty envelope into prediction-based TE optimization. Below, we consider three cases and illustrate how they can be efficiently implemented.

In the first case, we consider a penalty envelope on the absolute value of MLU on the set of possible traffic demands (*i.e.*, the set $\mathcal{X}$) which satisfy access capacity constraints. The problem can be formulated as follows:

$$\begin{array}{rll} \min & o(f,\mathcal{D}) & (7) \\ \text{subject to} & f \text{ is a routing;} & \\ & \forall \text{ links } l, \forall d_{ab} \ge 0 \text{ such that} & \\ & \sum_{b\in V} d_{ab} \le R_a^{OUT}, \sum_{b\in V} d_{ba} \le R_a^{IN}, & (8) \\ & \sum_{ab} d_{ab} f_{ab}(l)/c(l) \le \bar{r}, & \end{array}$$

where R_a^{OUT} and R_a^{IN} are the aggregated capacities of inbound and outbound access links of node a, respectively; they have equal values when access links have equal capacities in both directions. In this formulation, $\bar{r}$ is an upper bound on MLU.

The constraints (8) are not standard LP formulation. However, they can be tested by solving the following "slave LP" for each link l, and testing whether or not the objective is $\le \bar{r}$.

$$\begin{array}{rll} \max & \sum_{a,b} f_{ab}(l) d_{ab}/c(l) & (9) \\ \text{subject to} & \forall a,b \in V : d_{ab} \ge 0, & \\ & \sum_{b\in V} d_{ab} \le R_a^{OUT}, \sum_{b\in V} d_{ba} \le R_a^{IN}. & \end{array}$$

Using linear programming duality, we can show that the objective of (9) is $\le \bar{r}$ if and only if the following set of constraints can be satisfied:

$$\begin{array}{l} \forall a \in V : \mu_l(a) \ge 0, \nu_l(a) \ge 0; \\ \forall a,b \in V : f_{ab}(l)/c(l) \le \mu_l(a) + \nu_l(b); \\ \sum_{a\in V} (\mu_l(a) R_a^{OUT} + \nu_l(a) R_a^{IN}) \le \bar{r}. \end{array}$$

Figure 3: LP constraints to provide MLU penalty envelope over the set of demands satisfying access capacity constraints.

The variables $\mu_l(a)$ and $\nu_l(a)$ are dual multipliers on the node capacity constraints R_a^{OUT} and R_a^{IN}, respectively.

We can then replace the constraints (8) with the set of constraints in Figure 3. This gives an TE optimization problem with MLU penalty envelope over the set of traffic demands satisfying access capacity constraints.

There can be scenarios where it is more convenient to use a penalty envelope based on performance ratio than on the absolute value of MLU (*e.g.*, the feasible MLU envelope is too high). In the second case, we consider how to handle a performance-ratio penalty envelope on a convex set $\mathcal{X}$ formed as the convex hull of a set of TMs $\{D_1,\ldots,D_H\}$:

$$\begin{array}{lll} \min & o(f,\mathcal{D}) & (10) \\ \text{subject to} & f \text{ is a routing;} & \\ & \forall \text{ links } l, \forall \text{ TM } D = \sum_{h=1}^{H} t_h D_h, t_h \geq 0, \sum_{h=1}^{H} t_h = 1, & \\ & \frac{\sum_{ab} d_{ab} f_{ab}(l)}{c(l)} \leq \bar{r}\cdot \mathrm{OU}(D). & (11) \end{array}$$

Note that in the preceding problem formulation, there is one constraint for each traffic matrix $D \in \mathcal{X}$. Since the number of such matrices are infinite, this is not a standard linear programming problem. To solve the problem, we observe that the performance ratio $P(f,D)$ is *scale-free*:

$$P(f,D) = P(f,\alpha D), \quad \text{for all scalar } \alpha > 0. \tag{12}$$

Then we have the following result:

LEMMA 1. *Computing the performance-ratio optimal routing over the convex hull is equivalent to computing the performance-ratio optimal routing over a convex cone with the additional constraint $OU(D) = 1$.*

PROOF. For any $\mathcal{D}$, by the scale-free property of performance ratio $P(f,D)$, we have that:

$$\begin{aligned} P(f,\mathcal{D}) &= \max_{D\in\mathcal{D}} P(f,D) \\ &= \max_{D\in\mathcal{D}} P(f,\frac{D}{OU(D)}) \\ &= \max_{D\in\mathcal{D}} U(f,\frac{D}{OU(D)}) \quad (\because OU(\frac{D}{OU(D)}) = 1) \\ &\overset{D'=\frac{D}{OU(D)}}{=} \max_{D':\exists D\in\mathcal{D}, D'=\frac{D}{OU(D)}} U(f,D') \\ &= P(f,\mathcal{D}'), \end{aligned}$$

where the set $\mathcal{D}' = \left\{D' : \exists D \in \mathcal{D}, D' = \frac{D}{OU(D)}\right\}$.

Apply the preceding result when $\mathcal{D}$ is a convex hull, *i.e.*, $\mathcal{D} = \{D : D = \sum_{h=1}^{H} t_h D_h, t_h \geq 0, \sum_{h=1}^{H} t_h = 1\}$, it can be shown that the corresponding $\mathcal{D}' = \{D' : D' = \sum_{h=1}^{H} t_h D_h, t_h \geq 0, OU(D') = 1\}$ is a convex cone. This proves the lemma. □

Applying Lemma 1 to $\mathcal{X}$ (instead of $\mathcal{D}$), we obtain the formulation in Figure 4.

The formulation in Figure 4 still involves infinite number of constraints because it has one constraint for each D in the convex cone. However, the last two lines of constraints in Figure 4 can be tested by solving, for each link l, the following "slave LP", and testing if the objective is $\leq \bar{r}$.

$$\begin{array}{lll} \max & \sum_{a,b} f_{ab}(l) d_{ab}/c(l) & (13) \\ \text{subject to} & g_{ab}(e) \text{ is a flow of demand } d_{ab}; & \\ & \forall \text{ link } m, \sum_{a,b} g_{ab}(m) \leq c(m); & \\ & \forall a,b, d_{ab} = \sum_{h=1}^{H} t_h d_{ab}^h, t_h \geq 0. & (14) \end{array}$$

$$\begin{array}{ll} \min & o(f,\mathcal{D}) \\ \text{subject to} & f \text{ is a routing;} \\ & \forall \text{ link } l, \forall \text{ TM } D = \sum_{h=1}^{H} t_h D_h, t_h \geq 0, OU(D) = 1; \\ & \sum_{ab} d_{ab} f_{ab}(l)/c(l) \leq \bar{r}. \end{array}$$

Figure 4: Optimal TE with a performance-ratio penalty envelope over a convex set.

Using an approach similar to that in [7], we can show by linear programming duality, that the objective of (13) is $\leq \bar{r}$ if and only if the following set of constraints can be satisfied:

$\forall$ links $l,m : \pi(l,m) \geq 0$;
$\forall$ link l, nodes $i,j : p_l(i,j) \geq 0$, with $p_l(i,i) = 0$;
$\forall$ link $l : \sum_m \pi(l,m)c(m) \leq \bar{r}$;
$\forall$ link l, OD pair $a \to b : f_{ab}(l)/c(l) \leq p_l(a,b) - \lambda_l(a,b)$;
$\forall$ link l, node i, link $m = (j,k) : p_l(i,k) \leq p_l(i,j) + \pi(l,m)$;
$\forall$ link l, $h = 1,\ldots,H : \sum_{a,b} \lambda_l(a,b) d_{ab}^h \geq 0$.

Figure 5: LP constraints to provide performance-ratio penalty envelope over a convex set.

Compared with the LP models developed in [7], our model has additional Lagrange multipliers $\lambda_l(a,b)$, which correspond to the conic combination constraint (14).

We can then replace the last two lines of constraints in Figure 4 with the set of constraints in Figure 5, and form an TE optimization problem with performance-ratio penalty envelope over a set of possible traffic demands expressed as a convex-hull. When the TE optimization objective in (10) is the performance-ratio function over a convex set $\mathcal{D}$, we can similarly derive LP constraints.

The third case is a special case of the preceding case when the set of possible traffic matrices includes all non-negative traffic demands. The restriction imposed by the penalty envelope requirement in this case can be incorporated as a set of linear constraints. Specifically, a routing f has performance-ratio penalty envelope $\bar{r}$ if and only if the constraints in Figure 6 are satisfied.

$\forall$ links $l,m : \bar{\pi}(l,m) \geq 0$;
$\forall$ link l, nodes $i,j : \bar{p}_l(i,j) \geq 0$, with $\bar{p}_l(i,i) = 0$;
$\forall$ link $l : \sum_m \bar{\pi}(l,m)c(m) \leq \bar{r}$;
$\forall$ link l, OD pair $a \to b : f_{ab}(l)/c(l) \leq \bar{p}_l(a,b)$;
$\forall$ link l, node i, link $m = (j,k) : \bar{p}_l(i,k) \leq \bar{p}_l(i,j) + \bar{\pi}(l,m)$.

Figure 6: LP constraints to provide performance-ratio penalty envelope over the set of all non-negative traffic demands.

4. EVALUATIONS

In this section, we evaluate our algorithms.

4.1 Evaluation Methodology

Dataset description: We evaluate our algorithms using both real and synthetic data. We use the real topologies and traffic demand

Network	Aggregation level	#Nodes	#Links	Oblivious ratio	Penalty envelope
US-ISP	PoP-level	-	-	2.045	2.50
Abilene	router-level	11	28	1.853	2.00
Abovenet	PoP-level	15	60	2.014	2.05

Table 1: Summary of network topologies used.

Network	Description	Period studied	Time interval
US-ISP	Tier-1 ISP hourly traffic traces	a month in 2005	hourly traffic count
Abilene	Netflow data collected every 5 minutes from a number of universities and enterprises on the Internet-2	03/01/04 - 09/10/04	5-minute traffic count

Table 2: Summary of real traffic traces used.

matrices from Abilene and a major tier-1 ISP in US (anonymized as US-ISP). For Abilene, we collected the router-level topology and 6-month worth of traffic demands from Abilene Observatory [2]. Our dataset is available at [1]. For US-ISP, we rely on a PoP-level model, which differs from the real router-level topology and traffic demands, yet still illustrates the scope and power of the methods proposed here. In addition, we use the PoP-level Abovenet topology inferred by RocketFuel [35]. We use OC192 as the capacity for links in the Abovenet topology, and use the gravity model [53] described in [33] to generate synthetic traffic demands. Table 1 summarizes the topologies in use, their oblivious ratios, and the default performance-ratio penalty envelopes used in evaluation. Table 2 summarizes the real traces in use. Note that for proprietary reasons, we omit the numbers of nodes and links of US-ISP in Table 1.

Performance metrics: We use the following two performance metrics to compare different algorithms: (i) maximum link utilization (MLU) as defined in Equation (2); and (ii) performance ratio as defined in Equation (4). For both metrics, lower values indicate more efficient resource utilization, and hence are preferred. Note that the MLU defined in Equation (2) allows utilization to be above 100% when the traffic demand is large, while in practice, link utilization cannot exceed 100%. To be consistent with terminologies used by other authors, we use MLU with the understanding that it means traffic intensity and can exceed 100%. Also note that in our real-trace-based evaluations, we assume that traffic demands do not change as a function of the performance of a TE algorithm. When a TE algorithm performs badly and leads to network overload, TCP flows might be able to react and reduce network demand, resulting in lower demand than that happened in the real traces. However, this leads to reduced network throughput.

Algorithms: We categorize the algorithms we evaluate into the following three classes: (i) oblivious routing, (ii) prediction-based TE, and (iii) COPE. For all algorithms, we compute the routing by solving the corresponding linear programs using CPLEX [18]. By default, CPLEX uses dual simplex method to solve linear programs, but this is a poor choice for COPE. Given the particular structure of our problem formulation, we use the barrier method without crossover [12].

- Oblivious routing (*oblivious*): We compute the routing that gives the optimal oblivious ratio using the algorithm described in [7].

- Prediction-based TE: We evaluate the following three algorithms in this category. The first one requires routing update every time interval (*i.e.*, 1 hour in US-ISP, and 5-minutes in Abilene), whereas the latter two require routing update once a day. We have also conducted experiments using the Abilene traces, with predictions based on traffic of the preceding week and routing update every week, and the results are similar.
  - Dynamic (*dynamic*): At the beginning of each interval, we compute the optimal routing based on the traffic demand in the previous interval, and apply it to the traffic in the current interval. This algorithm models online traffic engineering (*e.g.*, [19, 26]). Note that the change in traffic demand from the previous interval can result in less efficient routing, compared to the optimal for the current interval.
  - Peak traffic demand (*peak*): At the beginning of each day, an optimal routing is computed based on the traffic demand in the peak interval (in terms of the total volume of traffic) of the previous day (for US-ISP) or the previous day and the same day of the previous week (for Abilene). This scheme has been suggested by [4].
  - Multiple traffic matrices (*multi*): At the beginning of each day, an optimal routing is computed based on a set of traffic matrices collected. We use the previous day's traffic matrices to compute routing for US-ISP, and use the traffic matrices in the previous day and the same day of the previous week to compute routing for Abilene. This algorithm is used in [34]. In addition, we also evaluate its variant suggested in [49], which selects traffic matrices of 6 consecutive intervals with the highest total volume of traffic. The results are similar and omitted for the interest of brevity.

- COPE: We evaluate two versions of COPE. For both of these two versions, we optimize routing based on the convex hull constructed from the set of traffic matrices collected from the previous day (for US-ISP), or from the previous day and the same day last week (for Abilene), subject to a performance-ratio penalty envelope on all non-negative traffic demands. Both versions require routing update only once a day, and are cheap to implement. The two versions differ only in their objective functions.
  - Minimizing the performance ratio (*COPE-ratio*): To optimize for performance ratio as the objective function over a convex set, we adopt the technique developed in Section 3.4 to handle penalty envelope on a convex set. Note that we also refer to COPE-ratio as COPE for short.
  - Minimizing MLU (*COPE-MLU*): This version of COPE is a variation of COPE-ratio which uses the absolute value of MLU, instead of performance ratio, as its objective function. Note that we can consider COPE-MLU as an extension of *multi* with penalty envelope.

4.2 Evaluation Results

US-ISP: First, we evaluate the performance of different algorithms using US-ISP, a large tier-1 US ISP. For confidentiality, we cannot report absolute MLU for US-ISP. Instead, we report relative MLU normalized by $MLU_{opt,max}$, where $MLU_{opt,max}$ is the highest maximum link utilization under the optimal routing over the entire month. Note that there are a small fraction of intervals without traffic demand information in the trace and we exclude those intervals in our evaluations.

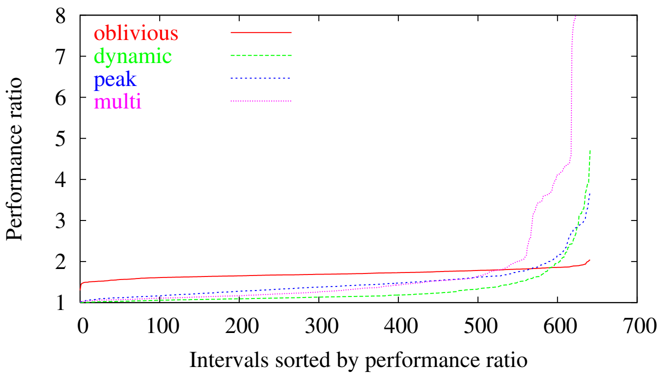


Figure 7: Oblivious vs. prediction-based TE: US-ISP traces. The performance ratio is truncated at 8.

We first compare the algorithms using performance ratio as the metric. Figure 7 compares oblivious routing with prediction-based TE. It plots the performance ratio versus the time interval sorted based on the performance ratio. Each time interval spans 1 hour. On one hand, we observe that prediction-based TE out-performs oblivious routing in most cases. For example, the performance ratio under the prediction-based algorithms is less than 1.2 for about half of the time intervals. Among the prediction-based algorithms, dynamic performs the best in the common case. This is expected as it updates every interval, while the others update every day. In comparison, the performance ratio of oblivious routing is above 1.5 for almost all of the intervals. On the other hand, the largest performance ratio under oblivious routing is 2.0355, whereas the largest performance ratios of prediction-based algorithms are all above 3.5, with multi even goes to higher than 8. Note that we limit the y-axis to 8 so that we can easily see the difference under normal traffic. The prediction-based algorithms incur large performance ratios because the inaccurate traffic prediction makes them perform significantly worse than the optimal. These results indicate that prediction-based TE is good at optimizing common-case performance, but suffers from large performance penalty when traffic demands change significantly. On the other hand, oblivious routing is good at handling unexpected traffic, but suffers on common-case performance. Neither class of algorithms dominates the other.

Figure 8 plots the performance ratios of the COPE algorithms, oblivious routing, and the common-case best-performing dynamic algorithm. Note that in order to display the curves clearly, we have truncated the performance ratio of dynamic, whose performance ratio will shoot up well beyond 2.2. We observe that for the common case, COPE algorithms achieve performance ratio which is about 6% higher than the dynamic algorithm, and significantly out-perform oblivious routing; for the worst case (in the traces), COPE algorithms even out-perform oblivious, and significantly out-perform dynamic prediction-based routing. Thus, COPE achieves the best of both.

Next, we compare the algorithms using normalized MLU as the metric. Figure 9 shows time series plots of normalized MLU under

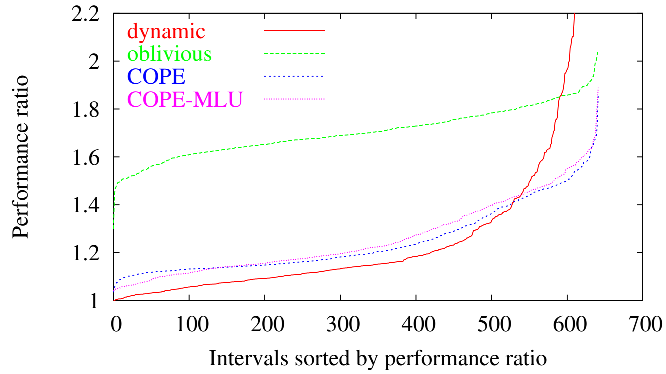
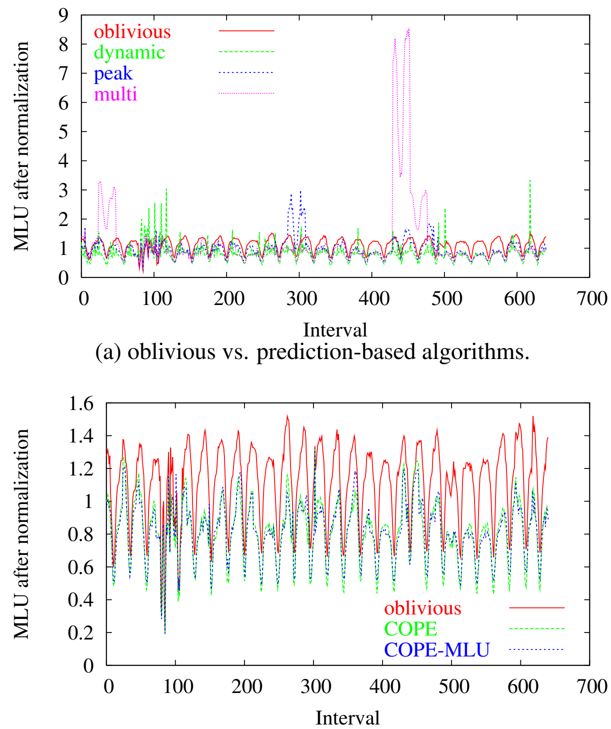


Figure 8: COPE vs. oblivious and dynamic: US-ISP traces. The performance ratio is truncated at 2.2.



(a) oblivious vs. prediction-based algorithms.

(b) oblivious vs. COPE.

Figure 9: Time series plots of normalized MLU: US-ISP traces.

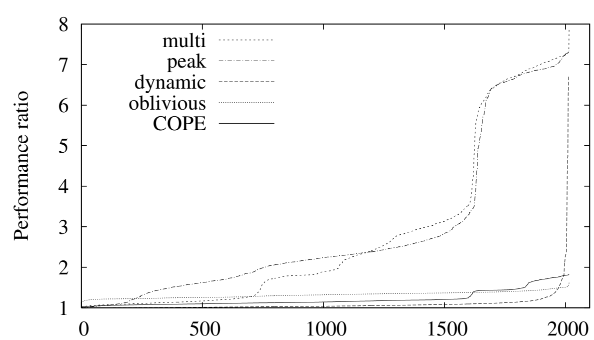


Figure 10: Comparison of performance ratio: Abilene traces from April 9 to April 15, 2004.

various routing schemes. From Figure 9(a), we can see that without penalty envelope, the prediction-based routing schemes can result in large spikes in normalized MLU. On the other hand, the normalized MLU of oblivious routing never exceeds 1.6, showing the robustness of oblivious routing. However, from Figure 9(b), we observe that the robust performance of oblivious routing is surpassed by COPE and COPE-MLU. They consistently yield lower link utilization than oblivious routing. We observe reduction in normalized MLU up to 56%.

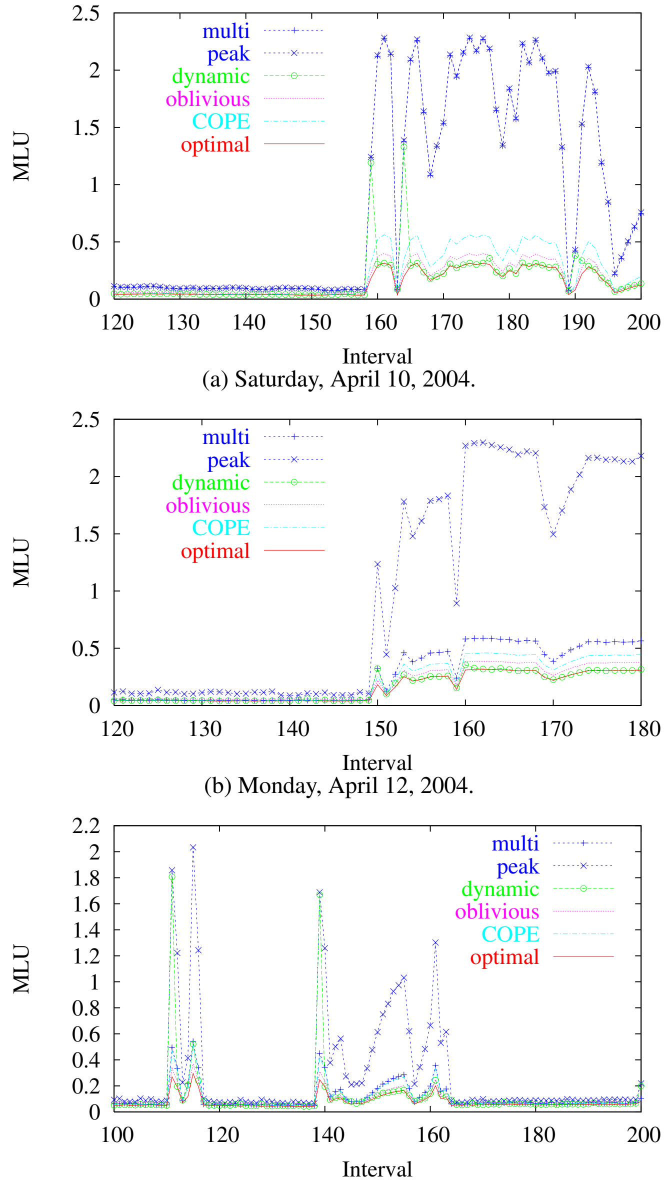
Abilene traces: The results of the US-ISP traces are normalized. Next we evaluate the algorithms using the public Abilene traces and show absolute values of MLU. Since COPE and COPE-MLU perform similarly, we report only COPE.

Figure 10 compares the algorithms in terms of performance ratio. It plots the performance ratio versus the time interval sorted based on the performance ratio. Each time interval spans 5 minutes. We make the following observations. First, predictions based on either multiple traffic matrices or peak traffic matrix perform poorly, with a performance ratio greater than 2 for about half of the intervals. Second, although dynamic routing achieves close-to-optimal performance ratio most of the time, the performance ratio can occasionally get as high as 6.7. Third, even though oblivious routing has better performance ratio on Abilene traces than on US-ISP traces, COPE still out-performs oblivious routing for about 80% of the intervals.

Figure 11 plots the MLU achieved by the algorithms during 3 days of the Abilene traces when traffic spikes happen. For clarity we zoom in to a few intervals in the day. We make the following three observations. First, on April 10, traffic engineering using multiple traffic matrices or peak matrix may drive the traffic intensity of the bottleneck link to be 240% of its link capacity (*e.g.*, at interval 161), while the optimal utilization is less than 50%. This high traffic intensity can lead to high packet loss rates and/or router crashes. Second, the dynamic algorithm exhibits interesting behavior. On April 10 and 14, it drives the traffic intensity of the bottleneck link to over 100% of its link capacity during some time periods (*e.g.*, 120% on April 10 at interval 159, and 181% on April 14 at interval 110). On the other hand, on April 12, it always achieves close-to-optimal performance. Closer examination of the traffic of the three days identifies that on April 12, there is a traffic ramp-up process, and thus the dynamic algorithm can adjust and performs well. However, on April 10 and 14, there is no such a ramp-up process before the spikes, and traffic changes rapidly; thus the dynamic algorithm overloads the network. Third, COPE and oblivious perform well under traffic spikes.

We next evaluate the performance of the algorithms when traffic is relatively steady and there are no traffic spikes. Figure 12 shows that for Friday, April 9, 2004 (all other days in the one-week period from April 9 to April 15 contain spikes). We make the following observations. First, during this normal-traffic day, the utilization of the network is low—the optimal MLU is only around 6%. This is typical of Abilene, which is over-provisioned. Second, the MLU achieved by using multiple traffic is about twice that of the optimal. This is because the traffic demands used for route computation do not match well with the real traffic, thus resulting in a large performance penalty. Third, the MLU achieved by oblivious routing is about 1.3 times optimal. This is better than the oblivious ratio of 1.853 for the Abilene network. However, a 30% performance penalty could still be large, especially if the network is heavily loaded. In comparison, the MLU of COPE is within 5% of optimal, and comparable to the dynamic algorithm under stable demands.

Abovenet: Next we evaluate the performance of COPE and oblivious routing using Abovenet topology. Since its traffic demands are



(a) Saturday, April 10, 2004.

(b) Monday, April 12, 2004.

(c) Wednesday, April 14, 2004.

Figure 11: Time series plots of MLU: Abilene traces.

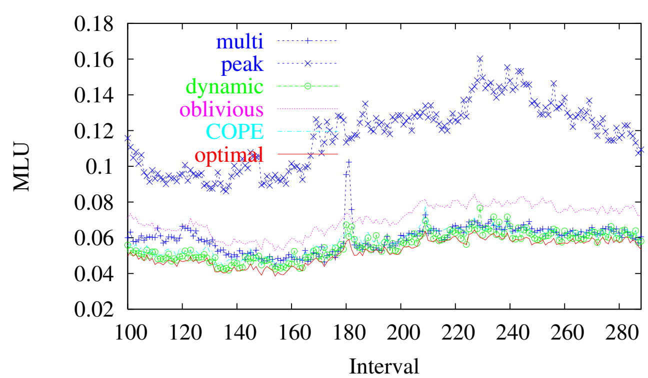


Figure 12: Time series plots of MLU: Abilene traces on Friday, April 9, 2004.

not available, we use the gravity model [53] described in [33] to generate sample synthetic traffic demands.

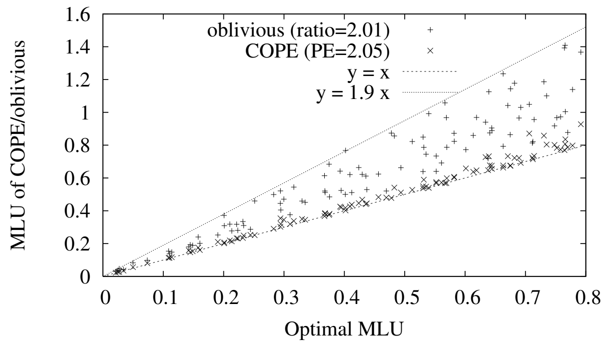


Figure 13: Scatter plot of MLU: Abovenet with gravity model traffic.

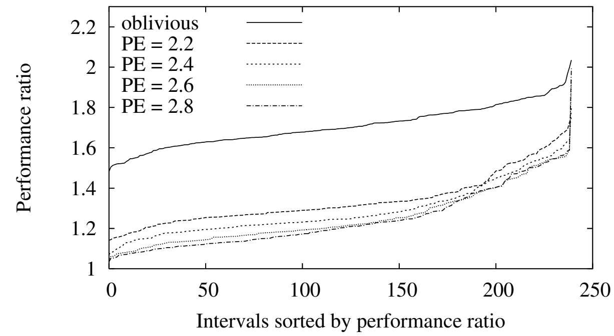
Figure 13 summarizes the results. The figure is a scatter plot, where each point represents one sample. The x-axis is the optimal MLU for the sample traffic matrix, and the y-axis is the MLU achieved by COPE or oblivious routing. In this evaluation, COPE uses a penalty envelope of 2.05, while the oblivious ratio of Abovenet is 2.014. In other words, the penalty envelope of COPE is only 2% more than the worst-case bound of oblivious routing. Interestingly, relaxing the worst-case bound slightly is sufficient to allow COPE to consistently achieve close-to-optimal performance. In comparison, the MLU of oblivious routing is much higher, up to 1.9 times of the optimal MLU.

Effects of penalty envelope:

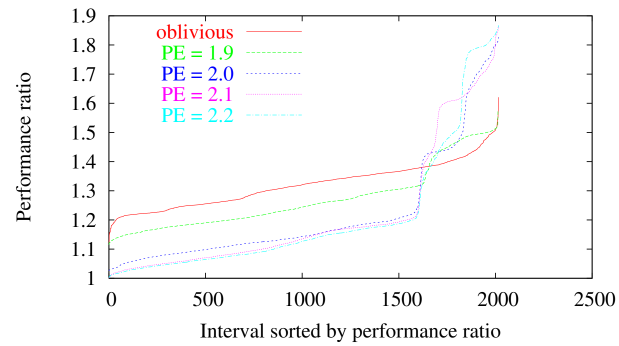
There is a trade-off in choosing different values of penalty envelope. When the value of the penalty envelope is high, the penalty guarantee is weak; however, a higher value of envelope leaves more room for optimizing common-case performance. When the envelope is set to be a very large value, the algorithm becomes prediction-based routing. On the other hand, when the value of the envelope is low, the penalty guarantee is strong; but not much room is left for optimizing common-case performance. When the penalty envelope is equal to the oblivious ratio of the whole possible traffic set, the scheme becomes oblivious routing.

To evaluate the effects of the penalty envelope, we study the performance of the US-ISP and Abilene traces as we vary the penalty envelope. Figure 14 shows the results. Note that the time interval of the US-ISP traces is an hour, while that of the Abilene traces is 5 minutes. Note also that we choose a 10-day period for the US-ISP traces, and choose a one-week period from April 9 to April 15 for the Abilene traces, as they best demonstrate the effects of penalty envelope.

We observe that as we increase the value of penalty envelope, the performance of most intervals improves. For example, for the US-ISP traces, the medium performance ratio (*i.e.*, the performance ratio at the middle of the sorted intervals) decreases by 23% as we increase the penalty envelope from 2.05 (the oblivious ratio) to 2.2; for the Abilene traces, the medium performance ratio decreases by 18% as we increase the penalty envelope from 1.85 (the oblivious ratio) to 2.0. When we increase the envelope beyond 2.2 for US-ISP and 2.0 for Abilene, however, the performance improvement is small. Since a larger value of penalty envelope implies larger potential worst-case performance ratio, 2.2 appears to be a good penalty envelope for US-ISP, and 2.0 for Abilene. In addition, Figure 14 shows that by choosing a penalty envelope slightly higher than the oblivious ratio, we achieve about 20% performance improvement for both US-ISP and Abilene. Note that US-ISP and



(a) US-ISP traces.



(b) Abilene traces: April 9-15, 2004.

Figure 14: Effects of penalty envelope.

Abilene have typical topologies that are representative for a wide range of networks. These results suggest that by increasing the penalty envelope to be just 10% above the lowest possible, we can already have enough room to optimize common-case performance.

Summary: Prediction-based algorithms can pay serious performance penalty when large traffic changes happen. They can drive traffic intensity to be over link capacity when the optimal network utilization is well below 100%. Oblivious routing pays performance penalty under normal traffic, driving utilization to be 30% to 90% higher than optimal in our evaluations. COPE avoids the performance penalties of both, and combines the best of both. Compared with the prediction-based algorithms, COPE retains their close-to-optimal performance under expected traffic demands, but avoids surge in network utilization and performance ratio under large traffic changes. Compared with oblivious routing, COPE also guarantees worst-case performance, but avoids the performance penalty of oblivious routing under predictable demands.

5. TRAFFIC ENGINEERING WITH INTERDOMAIN ROUTING

In this section, we develop a light-weight mechanism to handle interdomain routing. As we point out in Section 1, interdomain routing introduces two challenges. The first is point-to-multipoint demand; that is, there can be multiple equally-good egress points to a given external (*i.e.*, interdomain) destination prefix in the BGP decision process [17]. Thus, COPE needs to break the tie and assign the egress point of each external destination. Since egress links may become the bottlenecks of the network [16], this tie-breaking can affect the congestion of the network. Second, there are a large number of external destination prefixes each with their own set of egress points. ISPs have less control over interdomain routes, but some changes in the availability of interdomain routes can cause significant shifts of traffic. Thus, COPE should compute a routing that is robust to the traffic changes caused by changes of interdomain routes.

5.1 COPE with Interdomain Routing

Our high-level approach is a two-step process. In the first step, we convert the point-to-multipoint interdomain problem to a point-to-point intradomain problem in a topology extending the intradomain topology. By solving the latter problem, we obtain an assignment of external destinations to their egress points. This gives us point-to-point ingress-egress (IE) traffic matrices involving only intradomain nodes. In the second step, we apply COPE to the IE matrices to compute an intradomain routing. Since COPE tolerates traffic variations, the computed routing is robust against traffic variations due to shifts of egress points caused by dynamic interdomain routes.

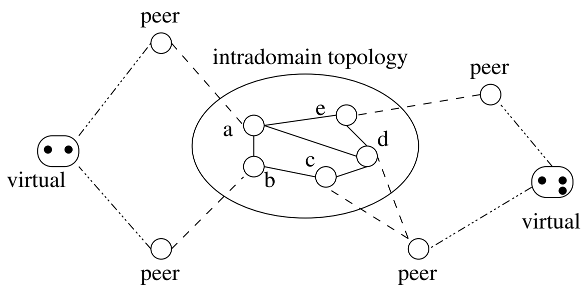


Figure 15: Extended graph G' which includes intradomain topology, peers, peering links (dashed links), and virtual nodes. Each virtual node represents a set of external destinations with the same set of egress peers.

Specifically, in the first step, we group external destination prefixes that share the same set of stable egress points into an equivalence class. Each equivalence class is represented by a virtual destination. We estimate the aggregated traffic demands to each virtual destination, and obtain origin-destination (OD) traffic demands. Then we construct an extended graph G', as illustrated in Figure 15, which includes the intradomain topology, the peers of the AS, the peering links, and the virtual destination nodes. We connect each virtual destination node with its corresponding peers using virtual links with infinite capacity. We apply COPE to the resulting topology G' to compute a robust routing. In particular, this routing will tell us how traffic to a virtual destination be split among its connected peers. Guided by the splitting ratios, we assign the egress point of each external destination prefix to approximate these ratios. This assignment of egress point can be implemented by BGP using local preference values. Figure 16 summarizes the algorithm.

In the second step, using the splitting ratios, we first compute ingress-egress (IE) traffic matrices involving only intradomain nodes. Then we use the IE matrices to compute a robust intradomain routing. To be robustness against possible large changes in traffic demands (*e.g.*, traffic change when a peer is down and many external destinations change egress points), we apply COPE to compute intradomain routing. The algorithm is shown in Figure 17. Note that unlike in Figure 16, here the inputs of COPE are the ingress-egress traffic matrices and the intradomain topology (which does not include peering links). This is important because we want to ensure that when peers or peering links go up and down, the penalty envelope of the resulted routing is not affected.

When the status of a peering link changes, the OD traffic that goes through this link is equally split over the alternative links that

1. Group traffic demands that share the same set of egress points into an equivalence class, EQ.
2. For each EQ, derive its pseudo OD demand that consists of all the OD demands belonging to EQ
3. Initialize G' to include intradomain topology, peers, peering links, and virtual nodes representing EQ's
4. Connect a virtual node to its corresponding peers using a virtual link with infinite capacity
5. Apply COPE to compute routing on G' for the pseudo demands
6. Derive the splitting ratios based on the routes
7. Assign interdomain prefixes to egress points

Figure 16: Step 1 of COPE with interdomain routing: convert point-to-multipoint to point-to-point demands by determining splitting ratios.

1. Initialize G to be the intradomain topology
2. Compute IE demands based on the splitting ratios derived in Figure 16.
3. Apply COPE to compute intradomain routes in G for ingress-egress demands

Figure 17: Step 2 of COPE with interdomain routing: compute robust intradomain routing.

are available for routing the OD demand. This change does not affect intradomain routing, since intradomain routing is robust to traffic changes through the use of penalty envelope.

5.2 Evaluation Results

Next we present our preliminary evaluation of COPE in dealing with interdomain routing dynamics.

We first extend the Abilene topology. From those ASes that peer with Abilene, we choose two (ESNET and DREN) and add them as virtual destinations. Each one represents a collection of interdomain destination prefixes. ESNET has 3 peering links with Abilene, and DREN has 4. For each peering link, we insert an AS in between. Therefore, in the final topology, Abilene has 3 neighboring ASes connecting to ESNET, and 4 connecting to DREN. The capacities of the peering links connecting Abilene to the inserted neighboring ASes are assigned using the actual peering link capacities. The oblivious routing ratio for this new topology is 2.039.

Next we determine the total demands from a router in Abilene to ESNET and DREN. Due to lack of good models for interdomain traffic matrices, we generate synthetic interdomain traffic demands using the Abilene traces as follows. We assume that a random portion (uniform from 0 to 0.5) of the traffic from a router in Abilene to the 3 border routers connecting to ESNET is actually targeted to ESNET. We similarly derive interdomain traffic demands to DREN.

We apply the algorithm described in Section 5.1 to the above derived topology and traffic demands. We set the penalty envelope to 2.2. We evaluate two failure scenarios: 1) a peering link connecting to ESNET (link 11) is down; and 2) both a link connecting to ESNET (link 11) is down, and a link connecting to DREN (link 15) is down.

Figure 18 shows the time series plot of MLU for Thursday, March 11, 2004. At the 145-th time interval, links 11 and 15 fail and remain failed until the end of the day. We include 4 curves in the plot: optimal with no link failures (*i.e.*, performance should the links not fail), optimal with the link failures, COPE without link failures, and COPE with the link failures. We observe that when the two links

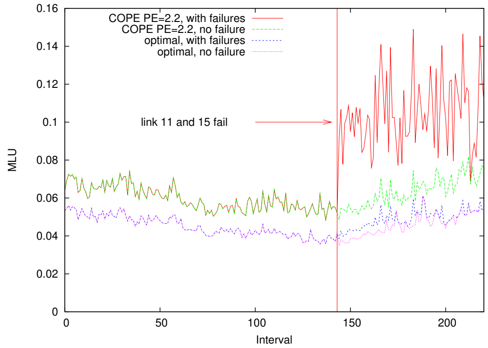


Figure 18: Times series plot of MLU: Abilene traces on Thursday, March 11, 2004.

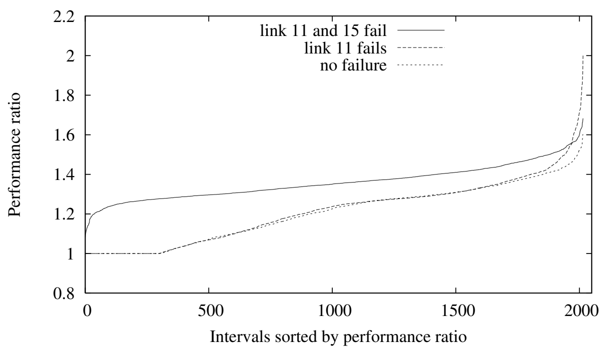


Figure 19: Scatter plot of sorted performance ratios: Abilene traces from March 8, 2004 to March 14, 2004.

fail, the MLU of COPE increases; but the performance penalty is small.

Figure 19 evaluates the performance ratio of COPE under link failures, for the time period from March 8, 2004 to March 14, 2004. It plots the sorted performance ratios of COPE under the following three scenarios: (i) there is no link failure, (ii) only link 11 is down, and (iii) both links 11 and 15 are down. We observe that when only link 11 is down, the performance degradation of the network under COPE is small. When two links are down, COPE has a higher performance ratio compared to the first two scenarios. However, even then the performance is mostly within 30% of optimal. Finally, we note that the performance ratios are always within the penalty envelope, and during most of the intervals, the performance ratios are well below the penalty envelope.

Summary: While preliminary, these results show that COPE is promising in handling interdomain route dynamics. As part of future work, we plan to further evaluate COPE in more diverse scenarios.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a novel scheme of traffic engineering: common-case optimization with penalty envelope (COPE), a class of traffic engineering algorithms that optimize for the expected scenarios while providing worst-case guarantee for unexpected scenarios. Using extensive evaluations based on real topologies and traffic traces, we show that COPE combines the best of prediction-based routing and oblivious routing, and can achieve efficient resource utilization under a wide variety of scenarios.

There are several avenues for future work. First, in this paper we focus on MLU. It is useful to apply COPE for optimizing other performance metrics. Second, we are interested in implementing COPE in a real network, and gain operational experience. Third, it will be interesting to find a way to dynamically detect that the common case traffic has changed and the network should recompute. Fourth, we believe that the general principle behind the design of COPE, optimizing for the common-cases while providing worst-case guarantee, can be applicable to many dynamic environments. We plan to explore other applications of this general principle.

7. ACKNOWLEDGMENTS

Hao Wang and Haiyong Xie are supported by NSF grants ANI-0238038 and CNS CNS-0435201. Yang Richard Yang is supported in part by NSF grants ANI-0238038. We are grateful to Pascal Bihler, David Goldenberg, Yanbin Liu, and Zheng Ma for their valuable comments. We are also grateful to the anonymous reviewers whose comments improve the paper.

8. REFERENCES

[1] Abilene topology and traffic dataset. http://www.cs.utexas.edu/~yzhang/research/AbileneTM/.

[2] Abilene Observatory. http://abilene.internet2.edu/observatory/.

[3] S. Agarwal, C.-N. Chuah, S. Bhattacharyya, and C. Diot. The impact of BGP dynamics on intra-domain traffic. In *Proceedings of Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, New York, NY, June 2004.

[4] S. Agarwal, A. Nucci, and S. Bhattacharyya. Measuring the shared fate of IGP engineering and interdomain traffic. In *Proceedings of the 13th International Conference on Network Protocols (ICNP) '05*, Boston, MA, Nov. 2005.

[5] R. Ahuja, T. Magnanti, and J. Orlin. *Network Flows*. Prentice Hall, 1993.

[6] D. Applegate, L. Breslau, and E. Cohen. Coping with network failures: Routing strategies for optimal demand oblivious restoration. In *Proceedings of Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, New York, NY, June 2004.

[7] D. Applegate and E. Cohen. Making intra-domain routing robust to changing and uncertain traffic demands: Understanding fundamental tradeoffs. In *Proceedings of ACM SIGCOMM '03*, Karlsruhe, Germany, Aug. 2003.

[8] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. *Overview and Principles of Internet Traffic Engineering, RFC 3272*, May 2002.

[9] D. O. Awduche. MPLS and traffic engineering in IP networks. *IEEE Communication Magazine*, pages 42–47, Dec. 1999.

[10] Y. Azar, E. Choen, A. Fiat, H. Kaplan, and H. Racke. Optimal oblivious routing in polynomial time. In *Proceedings of the 35th Annual Symposium on Theory of Computing*, 2003.

[11] N. Bansal, A. Blum, S. Chawla, and A. Meyerson. Online oblivious routing. In *Proceedings of ACM Symposium in Parallelism in Algorithms and Architectures (SPAA)*, 2003.

[12] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 2nd edition edition, 1999.

[13] D. Bertsekas and R. Gallager. *Data Networks*. Prentice-Hall, Second Edition, 1992.

[14] D. Bertsimas, D. Pachamanova, and M. Sim. Robust linear optimization under general norms. *Operations Research Letters*, 2004.

[15] T. Bressoud, R. Rastogi, and M. Smith. Optimal configuration for BGP route selection. In *Proceedings of IEEE INFOCOM '03*, San Francisco, CA, Apr. 2003.

[16] T. C. Bressoud, R. Rastogi, and M. A. Smith. Optimal configuration for BGP route selection. In *Proceedings of IEEE INFOCOM '02*, New York, NY, June 2002.

[17] M. Caesar and J. Rexford. BGP routing policies in ISP networks. *IEEE Network Magazine*, Nov. 2005.

[18] ILOG CPLEX: optimization software. http://www.ilog.com/products/cplex/.

[19] A. Elwalid, C. Jin, S. Low, and I. Widjaja. MATE: MPLS adaptive traffic engineering. In *Proceedings of IEEE INFOCOM '01*, Anchorage, AK, Apr. 2001.

[20] N. Feamster, J. Borkenhagen, and J. Rexford. Guidelines for interdomain traffic engineering. *ACM SIGCOMM Computer Communications Review*, Oct. 2003.

[21] N. Feamster and J. Rexford. Network-wide BGP route prediction for traffic engineering. In *Proceedings of ITCOM*, Boston, MA, Aug. 2002.

[22] N. Feamster, J. Winick, and J. Rexford. A model of BGP routing for network engineering. In *Proceedings of Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, pages 331–342, New York, NY, June 2004. ACM Press.

[23] V. Firoiu, I. Yeom, and X. Zhang. A framework for practical performance evaluation and traffic engineering in IP networks. In *IEEE ICT 2001*, 2001.

[24] B. Fortz, J. Rexford, and M. Thorup. Traffic engineering with traditional IP routing protocols. *IEEE Communication Magazine*, Oct. 2002.

[25] B. Fortz and M. Thorup. Internet traffic engineering by optimizing OSPF weights. In *Proceedings of IEEE INFOCOM '00*, Tel Aviv, Israel, Mar. 2000.
[26] S. Kandula, D. Katabi, B. Davie, and A. Charny. Walking the tightrope: Responsive yet stable traffic engineering. In *Proceedings of ACM SIGCOMM '05*, Philadelphia, PA, Aug. 2005.
[27] M. Kodialam, T. V. Lakshman, and S. Sengupta. Efficient and robust routing of highly variable traffic. In *Proceedings of Third Workshop on Hot Topics in Networks (HotNets-III)*, San Diego, CA, Nov. 2004.
[28] Y. Li, J. Harms, and R. Holte. A simple method for balancing network utilization and quality of routing. In *Proceedings of ICCCN*, San Diego, CA, 2005.
[29] Z. Ma, H. Wang, A. Krishnamurthy, A. Silberschatz, and Y. R. Yang. Achieving robust and optimal traffic engineering in current Internet. Technical report, Yale University, 2006.
[30] L. Qiu, Y. R. Yang, Y. Zhang, and S. Shenker. Selfish routing in Internet-like environments. In *Proceedings of ACM SIGCOMM '03*, Karlsruhe, Germany, Aug. 2003.
[31] H. Racke. Minimizing congestion in general networks. In *Proceedings of the 43rd Annual Symposium on Foudations of Computer Science*, Oct. 2002.
[32] J. Rexford, J. Wang, Z. Xiao, and Y. Zhang. BGP routing stability of popular destinations. In *Proceedings of Internet Measurement Workshop*, Marseille, France, Nov. 2002.
[33] M. Roughan. First order characterization of Internet traffic matrices. In *Proceedings of the 55th Session of the International Statistics Institute*, Sydney, Australia, Apr. 2005.
[34] M. Roughan, M. Thorup, and Y. Zhang. Traffic engineering with estimated traffic matrices. In *Proceedings of the Internet Measurement Conference*, Miami, FL, Oct. 2003.
[35] N. Spring, R. Mahajan, and D. Wetherall. Rocketfuel: An ISP topology mapping engine. Available from `http://www.cs.washington.edu/research/networking/rocketfuel/`.
[36] A. Sridharan, R. Guerin, and C. Diot. Achieving near optimal traffic engineering solutions in current OSPF/ISIS networks. In *Proceedings of IEEE INFOCOM '03*, San Francisco, CA, Apr. 2003.
[37] R. Teixeira, S. Agarwal, and J. Rexford. BGP routing changes: Merging views from two ISPs. *ACM SIGCOMM Computer Communications Review*, Oct. 2005.
[38] R. Teixeira, N. Duffield, J. Rexford, and M. Roughan. Traffic matrix reloaded: Impact of routing changes. In *Proceedings of Passive and Active Measurement*, Mar. 2005.
[39] R. Teixeira, T. Griffin, A. Shaikh, and G. Voelker. Network sensitivity to hot-potato disruptions. In *Proceedings of ACM SIGCOMM '04*, Portland, OR, Aug. 2004.
[40] R. Teixeira, A. Shaikh, T. Griffin, and J. Rexford. Dynamics of hot-potato routing in IP networks. In *Proceedings of Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, New York, NY, June 2004.
[41] S. Uhlig and O. Bonaventure. Implications of interdomain traffic characteristics on traffic engineering. In J. Crowcroft and A. Feldmann, editors, *Special issue on traffic engineering of European Transactions on Telecommunications*. 2002.
[42] L. G. Valiant. A scheme for fast parallel communication. *SIAM Journal on Computing*, 11(7):350–361, 1982.
[43] H. Wang, H. Xie, Y. R. Yang, L. E. Li, Y. Liu, and A. Silberschatz. Stable egress route selection for interdomain traffic engineering: Model and analysis. In *Proceedings of the 13th International Conference on Network Protocols (ICNP) '05*, Boston, MA, Nov. 2005.
[44] Z. Wang, Y. Wang, and L. Zhang. Internet traffic engineering without full mesh overlaying. In *Proceedings of IEEE INFOCOM '01*, Anchorage, AK, Apr. 2001.
[45] J. Wu, Z. M. Mao, J. Rexford, and J. Wang. Finding a needle in a haystack: Pinpointing significant BGP routing changes in an IP network. In *Proceedings of USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI '05)*, San Francisco, CA, May 2005.
[46] X. Xiao, A. Hannan, B. Bailey, and L. Ni. Traffic engineering with MPLS in the Internet. *IEEE Network Magazine*, pages 28–33, Mar. 2000.
[47] K. Xu, Z.-L. Zhang, and S. Bhattacharya. Profiling Internet backbone traffic: Behavior models and applications. In *Proceedings of ACM SIGCOMM '05*, Philadelphia, PA, Aug. 2005.
[48] T. Ye, H. Kaur, S. Kalyanaraman, K. Vastola, and S. Yadav. Optimization of OSPF weights using online simulation. In *Proceedings of International Workshop on Quality of Service (IWQoS))*, 2002.
[49] C. Zhang, Z. Ge, J. Kurose, Y. Liu, and D. Towsley. Optimal routing with multiple traffic matrices: Tradeoff between average case and worst case performance. In *Proceedings of the 13th International Conference on Network Protocols (ICNP) '05*, Boston, MA, Nov. 2005.
[50] C. Zhang, Y. Liu, W. Gong, J. Kurose, R. Moll, and D. Towsley. On optimal routing with multiple traffic matrices. In *Proceedings of IEEE INFOCOM '05*, Miami, FL, Apr. 2005.
[51] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker. On the constancy of Internet path properties. In *Proceedings of Internet Measurement Workshop 2001*, San Francisco, CA, Nov. 2001.
[52] Y. Zhang and Z. Ge. Finding critical traffic matrices. In *Proceedings of DSN '05*, Yokohama, Japan, June 2005.
[53] Y. Zhang, M. Roughan, C. Lund, and D. L. Donoho. An information-theoretic approach to traffic matrix estimation. In *Proceedings of ACM SIGCOMM '03*, Karlsruhe, Germany, Aug. 2003.
[54] R. Zhang-Shen and N. McKeown. Designing a predictable Internet backbone network. In *Proceedings of Third Workshop on Hot Topics in Networks (HotNets-III)*, San Diego, CA, Nov. 2004.

APPENDIX

A. SHORTEST-PATH-IMPLEMENTABLE INTRADOMAIN TE

The link-based routing computed by COPE can be implemented by path-based routing such as MPLS [29]. In this section, we seek a mechanism which enables us to compute routing that can be implemented by OSPF-style shortest path routing, with an appropriate set of positive link weights. Formally, A routing $f_{ab}(i,j)$ is *shortest-path-implementable* if there exists a set of positive link weights, such that any path p carrying positive flows for OD pair $a \to b$ is a shortest path from a to b.

From the above definition, it can be shown that a routing $f_{ab}(i,j)$ is shortest-path-implementable if and only if there exists a set of link weights $\{w(i,j)|(i,j) \in E\}$ and corresponding shortest distances $\{U(i,j)|i,j \in V\}$, such that:

$$\forall (i,j) \in E : w(i,j) \geq 1; \tag{15}$$
$$\forall a,i \in V, i \neq a : U(a,a) = 0, U(a,i) > 0; \tag{16}$$
$$\forall a \in V, (i,j) \in E : U(a,i) + w(i,j) \geq U(a,j); \tag{17}$$
$$\forall a,b \in V, (i,j) \in E :$$
$$f_{ab}(i,j) > 0 \Rightarrow U(a,i) + w(i,j) = U(a,j). \tag{18}$$

Given (15)-(18), if $f_{ab}(i,j) > 0$, then i is on the shortest path from a to j, and by induction i is also on the shortest path from a to b.

Let

$$SP(f,w,U) = \sum_{a,b \in V,(i,j) \in E} f_{ab}(i,j)[U(a,i) + w(i,j) - U(a,j)].$$

In view of (17) and the fact that $f_{ab}(i,j) \geq 0$, we have that $SP(f,w,U) \geq 0$ for any f, w and U satisfying (15)-(17), and that (18) is equivalent to $SP(f,w,U) = 0$. Therefore, if we can compute a routing f and its associated w, U such that $SP(f,w,U) = 0$, we can set link weights of OSPF-style routing protocols according to $w(i,j)$, which makes sure that traffic is always routed through shortest paths.

Specifically for COPE, we now introduce w and U as optimization variables and (15)-(17) as constraints. We can then add

$$\mathrm{SP}(f,w,U) = 0 \tag{19}$$

as a quadratic constraints to guarantee that the routing f computed is shortest path implementable. Alternatively, we can add

$$\alpha \cdot SP(f,w,U) \tag{20}$$

as a quadratic penalty term to the objective of COPE, where $\alpha > 0$. The added penalty term $\alpha \cdot SP(f,w,U)$ will ensure that the optimization not only minimizes the orginal objective of COPE, but also $SP(f,w,U)$. The scaler α is a very small positive number which is introduced to ensure that the minimization of the original objective of COPE takes higher priority. When this alternative approach is used, the routing computed is no longer guaranteed to be shortest path implementable. However, if the computed routing f, link weights w and corresponding shortest path distances U make $SP(f,w,U)$ small, the routing f is "nearly" shortest path implementable.