Towards Robust Multi-Layer Traffic Engineering: Optimization of Congestion Control and Routing

Jiayue He, Ma'ayan Bresler, Mung Chiang, and Jennifer Rexford

***Abstract*— In the Internet today, traffic engineering is performed assuming that the offered traffic is inelastic. In reality, end hosts adapt their sending rates to network congestion, and network operators adapt the routing to the measured traffic. This raises the question of whether the joint system of congestion control (transport layer) and routing (network layer) is stable and optimal. Using the established optimization models for TCP and traffic engineering as a basis, we find the joint system can be stablized and often maximizes aggregate user utility. We prove that both stability and optimality of the joint system can be guaranteed for sufficiently elastic traffic simply by tuning the cost function used for traffic engineering. Then, we present a new algorithm that adapts on a smaller timescale to changes in traffic distribution and is more robust to large traffic bursts. Uniting the network and transport layers in a multi-layer approach, this algorithm, Distributed Adaptive Traffic Engineering (DATE), jointly optimizes the goals of end users and network operators and reacts quickly to avoid bottlenecks. Simulations demonstrate that DATE converges quickly.**

***Index Terms*— Congestion control, Traffic Engineering, Network utility maximization, Optimization, Robustness, Routing.**

I. Introduction

IN THE Internet today, end hosts running the Transmission Control Protocol (TCP) adapt their sending rates in response to network congestion. Separately, network operators monitor their networks for signs of overloaded links and adapt the routing of traffic to alleviate congestion, in a process known as traffic engineering. TCP congestion control assumes that the network paths do not change, and traffic engineering assumes that the offered traffic does not change. Due to the layered network architecture, congestion control and routing operate independently, though their individual decisions are coupled. In this paper, we investigate whether the joint system is stable and optimal, then propose a good alternative system.

Traffic engineering and congestion control both solve, explicitly or implicitly, optimization problems defined for the entire network. Traffic engineering consists of collecting measurements of the traffic matrix—the observed load between each pair of entry and exit points—and performing a centralized minimization of a cost function that considers the resulting utilizations on all links (*e.g.*, [1], [2]). In contrast, TCP congestion control can be viewed as *implicitly* solving an optimization problem in a distributed fashion (*e.g.*, [3], [4], [5], [6]), where the many variants of TCP differ in the shape of user utility as a function of the source rate and in the type of Lagrange dual variables used.

Manuscript received April 15, 2006; revised Dec. 1, 2006. Parts of this material were presented at IEEE Globecom 2006 and poster session at IEEE Infocom 2006.

The authors are with Princeton University, USA (e-mail: {jhe, mbresler, chiangm, jrex}@princeton.edu).

Digital Object Identifier 10.1109/JSAC.2007.070602.

TABLE I
Differences between "TE Model" in Sections III-IV and "DATE" in Sections V-VI.

	TE Model	DATE
Focus	analysis	design
Approach	bottom-up	top-down
Timescale of route adaptation	offline	online
Computation of routes	centralized	distributed

Our study proceeds in two phases, first taking a "bottom up" approach that analyzes and characterizes the interaction between TCP congestion control and conventional traffic-engineering practices, followed by a "top down" approach where we design and evaluate a new, multi-layer, dynamic, distributed algorithm starting from a set of key design goals. The two systems differ in four ways, summarized in Table I.

For our "bottom up" approach, we use the established optimization models of congestion control and traffic engineering to study the interaction between them. We examine the following key questions through both analysis and simulation:

- **Stability**: Do the joint dynamics of congestion control and routing converge to an equilibrium?
- **Optimality**: If the joint system does converge, does the equilibrium maximize the aggregate user utility, over both the routing parameters and source rates?

Intra-domain routing can be abstracted as shortest-path routing based on link weights. Network operators compute optimal settings of the link weights in a centralized fashion based on a network-wide view of the offered traffic. Consistent with operational practices, our **Traffic Engineering model (TE Model)** has a distinct time-scale separation, where TCP converges under a fixed routing configuration, before any routing changes are made. Our model differs substantially from previous work that assumes each link weight is set locally, and dynamically, based on the congestion price [7], [8], [9]. Our simulation results show the TE Model is stable for a variety of topologies. In addition, the equilibrium point typically maximizes aggregate user utility. When link capacities have significant spread, however, the TE Model may deviate from this solution. These observations are rigorously explained where we prove that a modification to the cost function used in the TE Model can indeed guarantee stability and optimality (Theorems 1 and 2), but unfortunately at the expense of robustness.

To provide both performance and robustness, we then take a "top down" approach with the following design goals:

1) **Distributed:** in order to adapt on a small timescale, the algorithm should not require centralized computation.
2) **Robust:** the algorithm must be robust to traffic changes on a small timescale.
3) **Implementable:** for deployment to be feasible, only a limited number of changes to a limited number of routers should be required for the algorithm to run.
4) **Efficient:** the computations should be efficient enough to allow the routers to perform their other functions.

In developing an algorithm that jointly optimizes rates and routes in a way that satisfies all of the above design criteria, we first identify an objective function that balances the goals of end users and network operators, and then explore how to construct a stable, dynamic, distributed system that optimizes for this objective. In the resulting **Distributed Adaptive Traffic Engineering (DATE)** algorithm, edge routers compute the sending rate per source per path, and each link computes its effective capacity. Congestion price is fed back from the links to the sources to prevent the source rates from exceeding the *effective capacity*. On each link, we introduce *consistency price* to force the effective capacity to stay below the actual capacity. We prove that DATE is guaranteed to converge and jointly optimizes the goals of users and operators. Simulations further demonstrate that DATE converges fast and is robust with respect to parameter settings.

The following general conclusions are obtained as a result of our investigation of both the TE Model and DATE:

- **Confirming the intuition of network operators**: Our simulation results show the TE Model is stable for a variety of topologies (Section III).
- **Tension between performance and robustness**: A modification to the TE Model can guarantee stability and optimality (Theorem 1), but at the cost of robustness.
- **A dynamic, distributed algorithm can converge to a jointly optimal solution**: A stable, optimal, distributed algorithm exists (Theorems 2 and 3).

The rest of the paper is organized as follows. Section II introduces the network topology, routing model, congestion-control models and the TE Model. We first simulate the TE Model in Section III, and provide the analytic results and proofs in Section IV. Section V discusses the design goals for the joint system, and Section VI proposes, analyzes and simulates DATE. Section VII discusses related work, including a comparison of DATE with similar approaches in [10], [11], [12], [13], [14]. Section VIII concludes the paper and points to future work. Proofs of the two main theorems are presented in the Appendices.

II. Network Model

We focus on routing and congestion control in a single Autonomous System, where the operator has full view of the offered traffic load and complete control over routing, and a multipath routing model where traffic between source-destination pairs can be split arbitrarily across multiple paths. This is not the OSPF [15] or IS-IS [16] protocols used today, but can be implemented using MPLS [17]. While a session-level stochastic model is incorporated into our analysis of DATE, it is not explicitly considered for the TE Model, where we consider average TCP traffic profiles.

TABLE II
SUMMARY OF NOTATION USED IN SECTIONS II–IV

Symbol	Meaning
x_i	Rate of source i.
R_{li}	Fraction of traffic on link l for source i.
w_j^i	The fraction of source i on its j^{th} path.
c_l	Capacity of link l.
$U_i(x_i)$	Utility function for source i.
α	Parameterizing the TCP utility function.
$U_\alpha(x)$	Utility function modeling α-fairness.
β	Step size of the TCP algorithm.
u_l	Utilization of link l.
$f(u_l)$	Cost function.

Our notation follows the work in [7], [8]: in general, boldface are used to denote vectors and small letters are used to denote its components, *e.g.*, $\mathbf{x}$ with x_i as its i^{th} component; capital letters to denote matrices, *e.g.*, $\mathbf{R}$, or constants, *e.g.*, L and N. Superscript is used to denote vectors, matrices, or constants pertaining to source i, *e.g.*, $\mathbf{w}^i$ and $\mathbf{H}^i$. Also t is used to denote the iteration number, *e.g.*, $\mathbf{x}(t)$, in iterative algorithms. Table II presents a summary of the notation used in Sections II–IV.

A. Network Topology and Routing

A network is modeled as a set of L bidirectional links with finite capacities $\mathbf{c} = (c_l, l = 1, \ldots, L)$, shared by a set of N source-destination pairs, indexed by i; we often refer to a source-destination pair simply as "source i." [1]

We consider a routing model for best-effort packet-switched networks that closely reflects the operational practices in Internet Service Provider (ISP) backbones [1], [2]. We represent the current routing through a matrix R_{li} that captures the fraction of i's flow that traverses each link l; as such, we do not explicitly model the assignment of link weights, which has been explored in depth in previous work [1], [2]. The operators measure the offered load between each ingress-egress pair x_i. Based on the known network topology and the traffic matrix, the operators try to find the best routing matrix $\mathbf{R}$ to minimize network congestion[2].

For a given routing configuration, the utilization of link l is $u_l = \sum_i R_{li}x_i/c_l$. To penalize routing configurations that congest the links, candidate routing solutions are evaluated based on a cost function $f(u_l)$ that is strictly convex and increasing. In a recent comparison study [20], the cost function we consider is found to be the best network-wide traffic-engineering objective for a range of traffic conditions and performance metrics. The following optimization problem over $\mathbf{R}$, for fixed $\mathbf{x}$ and $\mathbf{c}$, captures the traffic-engineering practices:

$$\text{minimize} \quad \sum_l f(\sum_i R_{li}x_i/c_l). \tag{1}$$

[1]Index i here refers to a TCP session between two physical nodes in a topology where there could be multiple sessions between two physical nodes.

[2]By focusing on the operational practices in IP networks, our model differs substantially from earlier work on quality-of-service routing in connection-oriented networks (*e.g.* [18], [19] and references therein), where arriving connections are routed dynamically and each link performs admission control.

This optimization problem avoids solutions that operate near the capacity of the links and shifts flows to less utilized links where they can increase more freely. In practice, the network operators often use a piecewise-linear f for faster computation time [1], [2].

B. TCP Congestion Control

While the various TCP congestion-control algorithms were originally designed based on engineering heuristics, recent work, such as those surveyed in [5], [6], has shown through reverse engineering that they implicitly solve a convex optimization problem in a distributed fashion. Consider a network where each source i has a utility function $U_i(x_i)$ as a function of its total transmission rate x_i. The basic network utility maximization problem over source rate vector $\mathbf{x}$, for a given *fixed* routing matrix $\mathbf{R}$, is:

$$\begin{array}{ll} \text{maximize} & \sum_i U_i(x_i) \\ \text{subject to} & \mathbf{Rx} \preceq \mathbf{c}. \end{array} \tag{2}$$

The goal is to maximize aggregate user utility by varying $\mathbf{x}$ (but not $\mathbf{R}$), subject to the linear flow constraint that link loads cannot exceed capacity. TCP congestion-control algorithms implicitly solve (2), with different TCP variants maximizing different (increasing and concave) utility functions.

It is well-known that the utility functions can be picked based on several different grounds. First, a utility function can capture a user's degree of satisfaction with a particular throughput. Second, a utility function can be viewed as a measure of the elasticity of the traffic. Third, the aggregate utility captures the efficiency of the system in allocating bandwidth to the traffic. Fourth, some utility functions can lead to fair resource allocation. A particular family of widely-used utility functions is parameterized by $\alpha \geq 0$ [21]:

$$U_\alpha(x) = \begin{cases} \log x, & \alpha = 1 \\ (1-\alpha)^{-1} x^{1-\alpha}, & \alpha \neq 1. \end{cases} \tag{3}$$

Maximizing these α-fair utilities over linear flow constraints leads to rate-allocation vectors that satisfy the definitions of α-fairness in the economics literature.

The notion of α-fairness from [21] led to many TCP variants with different α-fairness interpretations. A utility function with $\alpha = 2$ was linked to TCP Reno. Through reverse engineering, TCP Vegas can be interpreted as $\alpha = 1$, as can STCP and FAST. XCP is shown to be maximizing for U_α as $\alpha \to \infty$ in the single-link case. One exception to this family of α-fair utility functions is TCP Tahoe, which has been reverse engineered to be maximizing the utility function $U(x) = \arctan x$ [5].

C. Traffic Engineering Model of Joint System

Our **TE Model of the joint congestion control and routing system** has two steps in a feedback loop, as shown in Figure 1. At time $t+1$, the congestion-control step computes new source rates based on the routing configuration from time t:

$$\mathbf{x}(t+1) = \text{argmax}_{\mathbf{x}} \sum_i U_i(x_i), \text{ subject to } \mathbf{R}(t)\mathbf{x} \preceq \mathbf{c}. \tag{4}$$

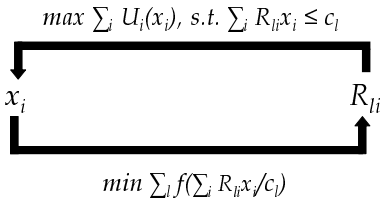


Fig. 1. A detailed view of the TE Model of joint congestion control and routing system.

Then the routing step computes new paths based on the source rates:

$$\mathbf{R}(t+1) = \text{argmin}_{\mathbf{R}} \sum_l f\left(\sum_i R_{li} x_i(t+1)/c_l\right). \tag{5}$$

The iterations of (4,5) repeat over time, with congestion control adapting the source rates to the new routes, and traffic engineering adapting the routes to the measured traffic.

III. Simulation of The TE Model

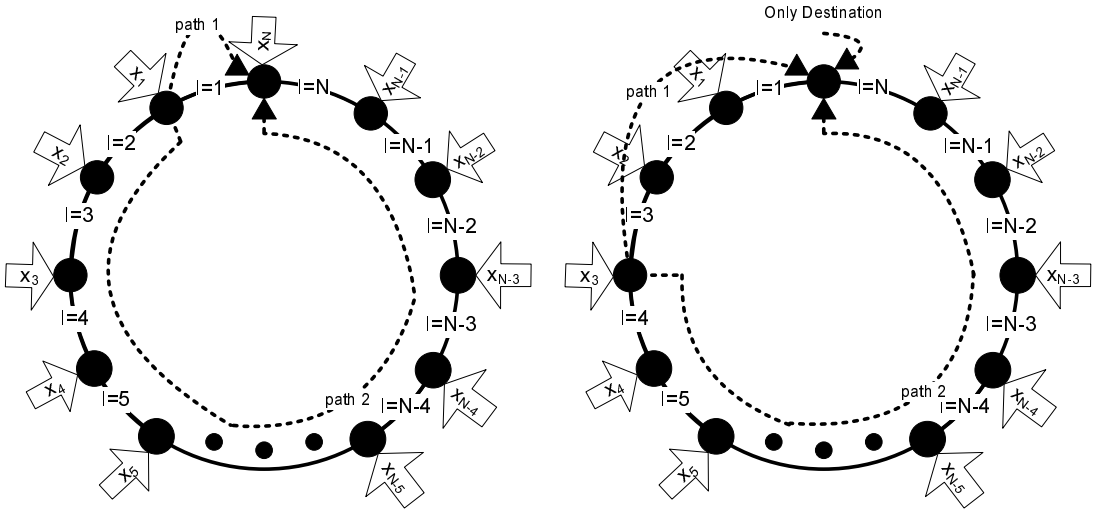
We first illustrate some interesting numerical observations before presenting theorems on stability and optimality in the next section. Our numerical experiments use a combination of the Matlab and MOSEK [22] environments.

A. Simulation Set-up

We evaluate two variants of TCP congestion control: $\alpha = 2$ (*e.g.*, TCP Reno) and $\alpha = 1$ (*e.g.*, TCP Vegas). For the cost-function $f(u_l)$, we use an exponential function, which is the continuous version of the function used in various studies of traffic engineering [1], [2].

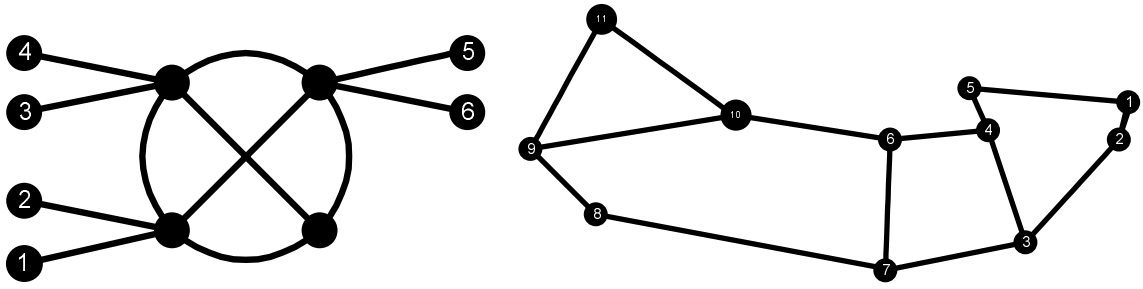
Our initial experiments evaluate a simple N-node ring topology, where we can easily scale the size of the network. To evaluate the influence of the traffic patterns, we consider two scenarios. In the first scenario, each node is a source sending to its clockwise neighbor; each source has two possible paths: a direct one-hop path and an indirect $(N-1)$-hop path. In the second scenario, node 1 is the destination and the remaining $N-1$ nodes are sources; each source x_i has an i-hop path and an $(N-i)$-hop path. Our experiments vary the number of nodes N and the capacity of link 1 (between nodes 1 and N).

To study realistic topologies with greater path diversity, we also experiment with the two networks in Figure 3. On the left is a tree-mesh topology, which is representative of a common network structure. In the middle is a full mesh representing the core of the network with rich connectivity. On the edge are three access tree subnetworks. Of the twelve possible source-destination pairs, $1-3$, $1-5$, $2-4$, $2-6$, $3-5$, and $4-6$ are chosen, and for each source-destination pair, the three minimum-hop paths are chosen as possible paths. On the right is the Abilene backbone network [23]. Of the many possible source-destination pairs, we choose $1-6, 3-9, 7-11$, and $1-11$. For each source-destination pair, we choose the four minimum-hop paths as possible paths. For the access-core and Abilene topologies, the simulations assume the link capacities follow a truncated (so as to avoid negative values) Gaussian distribution, with an average of 100 and a standard



(a) N nodes, N sources (b) N nodes, 1 destination

Fig. 2. Two N-node ring topologies with different traffic patterns.



(a) Access-Core topology (b) Abilene topology

Fig. 3. Two realistic topologies.

deviation that varies from 0 to 50. We simulate twenty random configurations for each value of the standard deviation. In all experiments, we start with an initial routing configuration that splits traffic evenly among the paths for each source-destination pair.

B. Suboptimality Gap Simulations

Given the structure of (2), it is natural to wonder if the interaction of congestion control and traffic engineering maximizes aggregate user utility. Previous work [12], [7] has proposed the following joint optimization problem:

$$\begin{array}{ll} \text{maximize} & \sum_i U_i(x_i) \\ \text{subject to} & \mathbf{Rx} \preceq \mathbf{c},\ \mathbf{x} \succeq \mathbf{0} \end{array} \tag{6}$$

where *both* $\mathbf{R}$ and $\mathbf{x}$ are variables.

Our experiments quantify the gap in aggregate utility between that at the equilibrium of the joint system and the optimal aggregate utility of (6). Such a gap between the achieved utility and the maximum utility signifies a loss in user satisfaction, and often implies also a loss in fairness or efficiency. Table III summarizes the key observations from the numerical experiments.

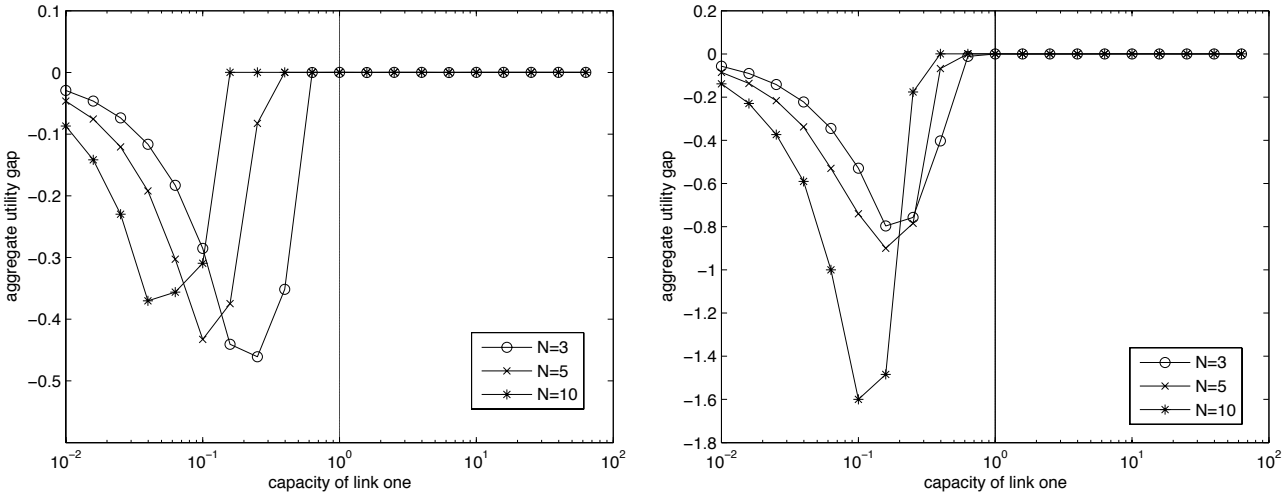
In Figure 4, we vary the capacity of link 1 and plot the gap in aggregate utility for ring topologies with three, five, and ten nodes, where each node communicates with its clockwise neighbor. The two graphs plot results for $\alpha = 1$ (*e.g.*, TCP Vegas) and $\alpha = 2$ (*e.g.*, TCP Reno), respectively. The graphs show trends that are very similar across a range of topology sizes, suggesting that the number of sources alone does not have a significant influence on the suboptimality gap. Similarly, the two TCP variants lead to very similar results.

TABLE III

SUMMARY OF RESULTS ON SUBOPTIMALITY GAP.

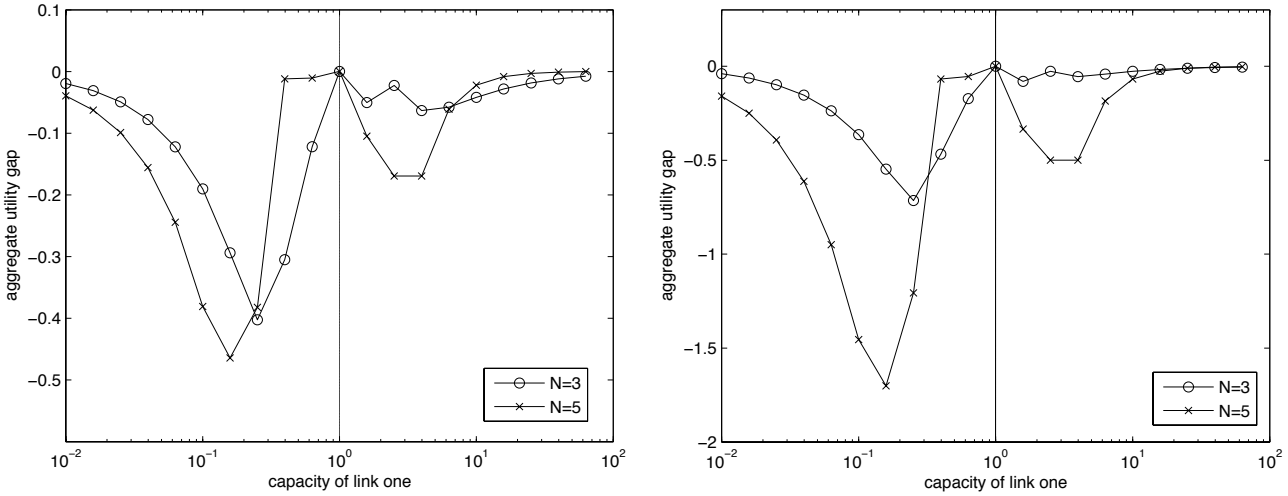
Figure(s)	Key Message
4a versus 5a	Traffic pattern can have a significant effect.
4a versus 4b 5a versus 5b	TCP variants give the same trend.
All figures	Relatively small suboptimality gap.
All figures	Homogeneity minimizes suboptimality gap.

The vertical line in the middle of the two graphs highlights the configuration where all links have unit capacity. The suboptimality gap is zero for a wide range of capacity configurations. When one link has much lower capacity than the other links, a suboptimality gap emerges. This occurs because the traffic-engineering step in the joint system stops making use of this low-capacity link, since the penalty for placing even a small amount of load on this link exceeds the cost of forcing the traffic on a longer path that places load on multiple links. When link 1 has an extremely low capacity, even the optimal solution cannot place much traffic on this link, leading to a small suboptimality gap.



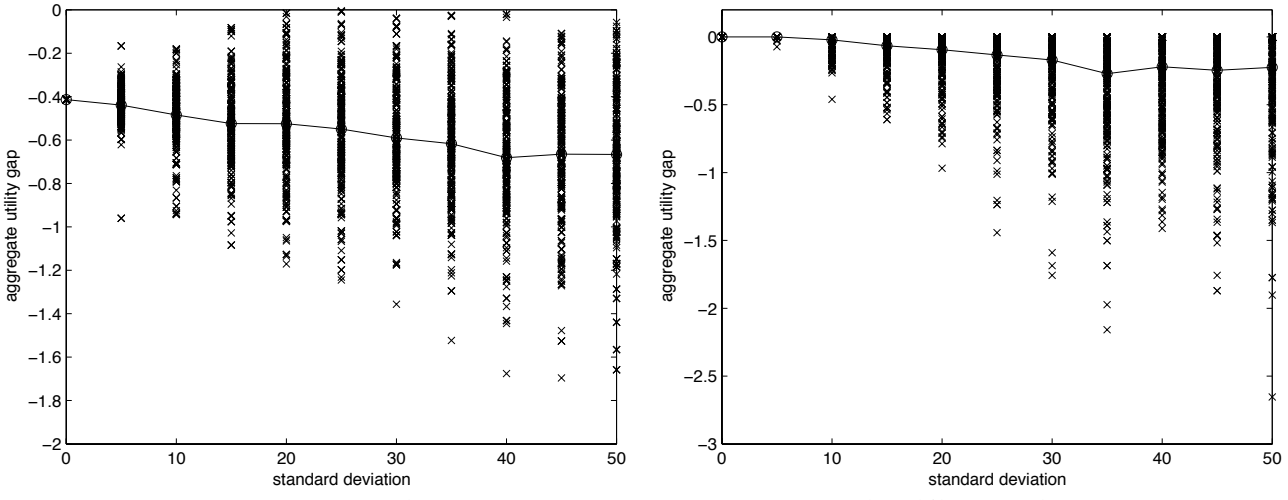
(a) $\alpha = 1$ (*e.g.*, TCP Vegas) (b) $\alpha = 2$ (*e.g.*, TCP Reno)

Fig. 4. Aggregate utility gap for the N-node, N-source ring.



(a) $\alpha = 1$ (*e.g.*, TCP Vegas) (b) $\alpha = 2$ (*e.g.*, TCP Reno)

Fig. 5. Aggregate utility gap for the N-node, 1-destination ring.



(a) Access-core topology (b) Abilene topology

Fig. 6. Aggregate utility gap for two realistic topologies (with $\alpha = 1$). A -x- marker denotes an individual test point and a -o- marker denotes the average.

The graphs in Figure 5 confirm that variations in link capacities affect the suboptimality gap. These graphs evaluate the N-node ring with one destination node, for two values of N and two TCP variants. In contrast to Figure 4, having

either a smaller or a larger capacity on link 1 leads to a suboptimality gap. This is not surprising because link 1 is a bottleneck link for this traffic pattern. If the link has a small capacity, the traffic-engineering step does not make use of the link, making the left part of these curves closely resemble the plots in Figure 4. If the link has a high capacity, the traffic-engineering step tries to direct more sources through the link; however, this is not the best solution when the capacity of link 1 is just slightly larger than the other links because traffic traverses longer paths, placing load on a larger number of links. Comparing Figures 4 and 5 illustrates the important role the traffic pattern plays in determining whether the joint system successfully maximizes aggregate utility.

The graphs in Figure 6 illustrate the effects of a variation in link capacities on realistic topologies. We show how the suboptimality gap depends on the standard deviation of the link capacities, which are all varied according to a truncated Gaussian distribution. We plot separate points for each of the 500 experiments for each value of standard deviation, as well as a curve that highlights the mean values. The trend that a more homogeneous capacity distribution (smaller standard deviation) leads to a smaller suboptimality gap exists, but it is much more subdued than in the ring topology and it is dominated by the variance. This suggests that, with realistic topologies, the relationship between link capacity and utility gap is more complex. One possible explanation is that the bottleneck link on each path is what matters, and, while that is easily correlated with varying a single link in the ring topology, the effect is coupled in a more complex topology. In addition, for the Abilene topology, a suboptimality gap exists even for a homogenous capacity distribution. While the results for the ring topology suggest that network operators could favor network configurations that enable (near) optimal solutions, the results for the access-core and Abilene networks suggest this may be quite challenging for more realistic topologies.

IV. Analysis of the TE Model

Our simulations showed that the TE Model (4,5) is stable and close to optimal for a range of topologies. We speculate, but have not yet shown it is provably stable for general topologies. In this section, we show how relaxing the constraint for (4) can lead to a *provably stable and optimal* joint system. By imposing conditions on f, the joint system can be *brought* arbitrarily close to optimality with respect to (6). This comes at the expense of robustness, however, and is not recommended for implementation.

Theorem 1: If (4) is replaced by the following unconstrained problem:

$$\mathbf{x}(t+1) = \text{argmax}_{\mathbf{x}} \sum_i U_i(x_i) - \sum_l f\left(\sum_i R_{li}(t)x_i/c_l\right), \quad (7)$$

then the TE Model converges to the optimum of

$$\text{argmax}_{(\mathbf{x},\mathbf{R})} \sum_i U_i(x_i) - \sum_l f\left(\sum_i R_{li}(t)x_i/c_l\right), \quad (8)$$

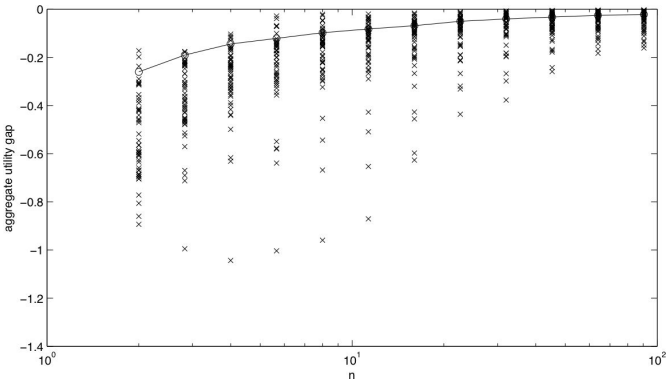


Fig. 7. Using the function $f(u_l) = nu_l^n$ in the TE model (5) and alternating iterations with TCP (4) results in a arbitrarily small gap in utility as $n \to \infty$.

for sufficiently concave utilities (i.e., sufficiently elastic traffic): $U_i''(x_i) \leq -\frac{U_i'(x_i)}{x_i}$. In particular, it converges for α-fair utilities when $\alpha \geq 1$ and for $\arctan$ utility of TCP Tahoe.

Outline of the Proof: The proof consists of three main steps. First we show that the joint *modified* congestion control and routing system is equivalent to a successive, alternating optimization over $\mathbf{x}$ and then $\mathbf{R}$ (i.e., a Gauss-Seidel type of algorithm) of (8). In particular, we need to modify (4) so that the congestion-control update corresponds exactly with the optimization over $\mathbf{x}$ in Gauss-Seidel algorithm. Then we provide a sufficient condition to guarantee convergence to the solution of (8). Finally the condition is examined for α-fair utilities and arctan utility. See Appendix I for the detailed proof. ■

Theorem 2: The cost function f can be chosen so that (8) is arbitrarily close to (6). In this asymptote, the TE model (4, 5) converges to the global optimum $(\mathbf{R}^*, \mathbf{x}^*)$ of (6).

Proof: From Theorem 1, the joint system (5,7) converges to (8). If (8) is arbitrarily close to (6), and (4) to (7) then (4, 5) will converge to the solution of (6).

By the penalty-function method (see [24] for details), there exists a penalty function P and a constant γ so that (6) is equivalent to (9):

$$\text{maximize}_{(\mathbf{x},\mathbf{R})} \sum_i U_i(x_i) - \gamma \sum_l P\left(\sum_i R_{li}x_i/c_l\right), \quad (9)$$

and (4) is equivalent to (7) provided that γ is sufficiently large, and P is convex, increasing, and zero for $\mathbf{Rx} \preceq \mathbf{c}$ (positive otherwise). Essentially, in (9) and (7), $-\gamma \sum_l P\left(\sum_i R_{li}x_i/c_l\right)$ in the objective function replaces the constraint $\mathbf{Rx} \preceq \mathbf{c}$ when γ is sufficiently large. If the operators choose a cost function f which is zero until $u_l = 1$, and sufficiently large afterwards, then it can match γP exactly. Such a function can be approximated as closely as desired, *e.g.* by choosing sufficiently large n in $f(u_l) = nu_l^n$ where $n > 0$ is a parameter that modulates the approximation accuracy with respect to the original TCP model. ■

Simulations confirm that the joint model converges arbitrarily closely to the optimum of (6) when the TE penalty function in (5) is replaced by $f(u_l) = nu_l^n$ and n is allowed to go to infinity. For example, Figure 7 plots the utility gap as a function of n for the Abilene topology with a standard deviation of 50. For larger values of n, the utility gap grows

arbitrarily small, in contrast to the large gap seen earlier in Figure 6(b). Although larger values of n narrow the optimality gap, we find that the joint system requires more iterations for convergence, leading to a sometimes substantial increase in convergence time. For simpler, more uniform topologies, we find that even small values of n suffice to close the optimality gap. These plots are omitted due to space limitations.

While modifying the traffic-engineering penalty function f will allow us to be arbitrarily close to the optimum of (6), it will also drive the network increasingly close to a solution with multiple links operating near capacity. This is a fragile point of operation for the network since a small burst in traffic would cause the traffic on certain links to exceed capacity. Once the traffic exceeds capacity, congestion is inevitable and so is the subsequent packet loss and delay increase. Here we have a trade-off between stability and optimality (with respect to (6)) on the one hand and robustness (with respect to short, high-volume traffic bursts) on the other.

V. Design Goals for Multi-layer Traffic Engineering

Although the TE model converges and performs well in practice, we believe that, for robustness, maximizing aggregate utility is not the most appropriate objective for the joint system. In addition, the centralized optimization of routing does not react on a small enough timescale to adapt to shifts in traffic (unlike the more adaptive TCP which, due to its decentralized nature, can operate on the time scale of packets).

In this section, we first propose an alternative objective for the joint system that captures the needs of users and operators alike. Then, we meet the challenges of a creating a dynamic and distributed algorithm that is guaranteed to converge to the global optimum of that objective.

A. Satisfying User and Operator Objectives

Theorem 1 shows modifying the TE Model causes the independent actions of end users and network operators to tend toward a solution that maximizes the aggregate user utility. While maximizing aggregate user utility seems like a very natural objective, and has been used in previous studies [12], [7], such a solution can be fragile to bursts in traffic. These observations hint that maximizing the aggregate user utility enhances performance of the individual users in the short term, but leaves the network as a whole fragile. One possible solution is to combine performance metrics (users' objective) with network robustness (operator's objective), leading to the following formulation as a joint optimization over $(\mathbf{x}, \mathbf{R})$:

$$\begin{array}{ll} \text{maximize} & \sum_i U_i(x_i) - \sum_l f(\sum_i R_{li} x_i / c_l) \\ \text{subject to} & \mathbf{R}\mathbf{x} \preceq \mathbf{c},\ \mathbf{x} \succeq \mathbf{0}. \end{array} \tag{10}$$

This objective favors a solution that provides both high aggregate utility and a low overall network congestion, to satisfy the need for performance and robustness.

In any optimization problem formulation, there are four-tuples: objective function, constraint set, variables, and constants. By having both $\mathbf{R}$ and $\mathbf{x}$ as optimization variables, we can obtain the best combination of source rate control and load balancing in alleviating bandwidth bottlenecks. In our problem, the constraint is that link load does not exceed capacity with the physical topology and the link capacities as constants. Finally, the objective function reflects a combination of source utility and operator cost. Solving this problem in a way that is temporally dynamic and spatially distributed is the goal of the rest of this paper.

B. Adapting Routing on a Smaller Timescale

Traffic engineering today is centralized, so each iteration requires at least a few minutes to collect accurate traffic statistics and perform the necessary computations. In practice, the offered traffic may change at a smaller timescale, so having a dynamic, distributed algorithm would be desirable. Yet previous research [25], [26], [27] has shown that load-sensitive routing alone is prone to instability. While TCP congestion control is an example of a stable, dynamic and distributed system, the combination of congestion control and adapative routing can result in an unstable system [7], [8]. Designing a stable system that adapts both source rates and routes requires care.

We introduce DATE (Dynamic Adaptive Traffic Engineering) to react on a fast timescale while ensuring stability. To prevent the typical oscillatory behavior, we take inspiration from the original TCP congestion-control algorithm while extending it to include traffic-engineering objectives. Like TCP congestion control, DATE uses congestion price to provide feedback about network conditions. To ensure network robustness, congestion price in DATE plays the role of ensuring DATE does not exceed the *effective capacity* rather than the actual capacity of the links. Each link updates its effective capacity to take into account a penalty imposed by traffic engineering. Finally, we introduce *consistency price* to ensure the effective capacity stays below the actual capacity.

The major *architectural* difference between DATE and traditional traffic engineering is that DATE shifts some of the management tasks directly into the routers. In today's traffic-engineering practices, the management system determines link weights in a centralized fashion, and link-weight based routing is done via OSPF or IS-IS in a distributed manner. For DATE, the management system is only responsible for choosing f and determining the Label Switched Paths (LSPs) needed to forward the traffic over multiple paths. In DATE routers are responsible both for determining how much traffic each path should carry and for forwarding the data packets on these paths. This is an alternative allocation of functionality between the network-management system and the routers.

VI. Distributed Adaptive Traffic Engineering

In this section, we describe the Distributed Adaptive Traffic Engineering (DATE) algorithm in Section VI-A, which is derived in Section VI-B from Equation (10). In Section VI-C, we evaluate DATE's convergence rate and robustness.

A. DATE Algorithm

DATE is a multipath routing protocol where the edge routers split traffic for each source-destination pair over multiple

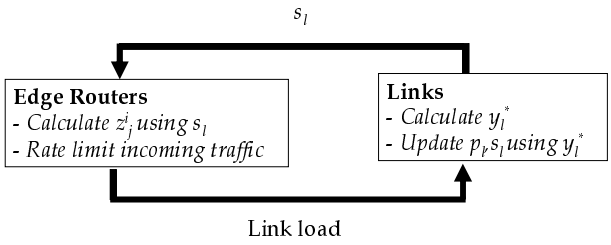


Fig. 8. A graphical view of DATE algorithm.

TABLE IV
SUMMARY OF NOTATION FOR DATE.

Symbol	Meaning
z_j^i	Rate of source i on its j^{th} path.
$\mathbf{H}$	Network topology as the set of all paths.
y_l	Effective capacity on link l.
s_l	Congestion price on link l.
p_l	Consistency price on link l.
β_s	Step size for update of congestion price.
β_p	Step size for update of consistency price.
T_p	Time it takes for feedback delay.

paths. In particular, the edge router computes a rate z_j^i for TCP session i on path j, and polices the incoming traffic to obey the rate limit. The key challenge in designing DATE is to determine how the edge routers should set the z_j^i values to ensure the resulting system is both stable and optimal. We illustrate the interplay between the edge routers (that compute the rates z_j^i) and the network links (that provide feedback s_l about network conditions) in Figure 8; the full list of notation is provided in Table IV.

The edge router updates z_j^i based on explicit feedback from the links, in the form of congestion prices s_l. In particular, the edge router maximizes the utility for TCP session i, while balancing the price of using path j. The path price is the product of the source rate with the price per load for path j (computed by summing s_l over the links in the path), as shown in Table V. This is very similar to the standard TCP dual algorithm in [5] except the local maximization is conducted over a *vector* $\mathbf{z}^i$, as opposed to only a scalar x_i, to capture the multipath nature of DATE. The parameter T_p represents the propagation delay for the edge router to receive the congestion-price information from the links.

The assignment of the z_j^i values at the edge routers determines the total traffic that traverses each link. The resulting load on link l is $\sum_j H_{lj}^i z_j^i$, which serves as implicit feedback that the link uses to compute the congestion price s_l. To avoid driving the link to full utilization, we base the congestion price on an *effective* capacity y_l that stays below the actual link capacity c_l. The equation for congestion price here is similar to [5], except for using the effective capacity rather than actual capacity; the congestion price is updated over time using a gradient method.

The effective capacity is kept below the actual capacity by a consistency price p_l that enforces the capacity constraint; as with s_l, the consistency price is updated over time using a gradient method. Then, the effective capacity y_l is updated per link using information from both prices and the cost function f. An economic interpretation is that the effective capacity balances the cost of using a link (represented by f) and revenue of using a link (represented by the product of the total price per load with the effective capacity). In this system, since all the constraints are linear, the prices are additive.

TABLE V
THE DATE ALGORITHM.

Edge Routers:

$$z_j^i(t+T_p) = \text{maximize}_{z_j^i} U_i(\mathbf{1}^T z^i) - z_j^i \sum_l s_l(t) H_{lj}^i$$

where z_j^i is the amount of load that TCP session i places on its j^{th} path where $x_i = \sum_j z_j^i$.

Links:

- Congestion price Update:

$$s_l(t+T_p) = s_l(t) - \beta_s \left(y_l(t) - \sum_i \sum_j H_{lj}^i z_j^i(t) \right),$$

  where β_s is the congestion price step size.
- Consistency price Update:

$$p_l(t+T_p) = [p_l(t) - \beta_p(c_l - y_l(t))]^+,$$

  where β_p is the consistency price step size. Since, $p_l \geq 0$, it must be mapped to a non-negative value.
- Effective Capacity Update:

$$y_l(t+T_p) = \text{minimize}_{y_l} f(y_l/c_l) - (s_l(t) + p_l(t)) y_l.$$

  where y_l is the effective capacity.

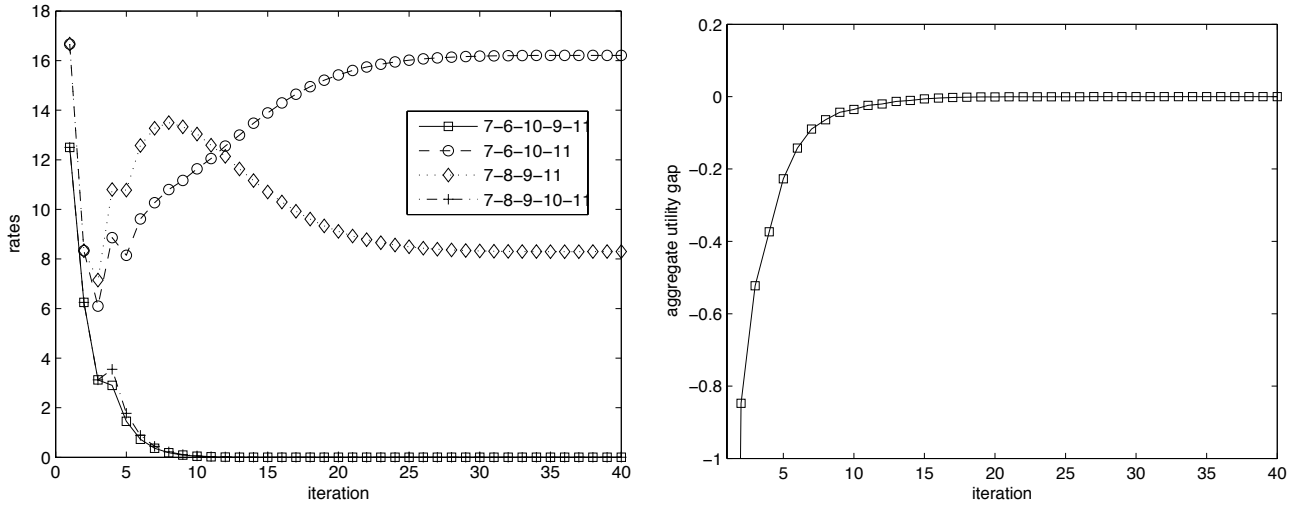
The computations at the edge routers are linear with the number of sources, while the computations at the link do not grow with the number of sources or the number of links. In addition to computation overhead, there are three new functionalities required by DATE that are not standard today. First, DATE will require MPLS for splitting traffic over multiple paths. Second, DATE will require frequent link-load measurements which is possible using the Simple Network Management Protocol (SNMP). Finally, DATE requires explicit rate limiting of the incoming traffic; this can be done by dropping packets sent above the allowed rate.

B. Stability and Optimality Results

In this subsection and Appendix II, we provide the analytical derivation and theoretical foundation of the DATE algorithm.

Theorem 3: The distributed algorithm DATE converges to the joint global optimum $(\mathbf{R}, \mathbf{x})$ of (10) for sufficiently small step sizes β_p and β_s.

Outline of the Proof: The key idea to arrive at a distributed algorithm is to decouple the coupled objective function (coupled over $\mathbf{R}$ and $\mathbf{x}$) in problem (10) by introducing an auxiliary variable and an additional constraint, and then use Lagrange dual decomposition to decouple both constraints. First we rewrite (10) as a convex problem by introducing new variable $\mathbf{z}$, which is a stacked vector with entries $z_j^i = x_i w_j^i$. Then we have introduced a new variable $y_l = \sum_i \sum_j H_{lj}^i z_j^i, \ \forall l$ to enable decoupling. Finally, we use dual decomposition and the gradient descent method to derive



(a) Source rate on each path (b) Aggregate utility

Fig. 9. Plots of source rate and aggregate utility versus time for source pair 7-11 with step sizes: $\beta_p = 2 \times 10^{-5}$, $\beta_s = 4 \times 10^{-5}$.

the DATE algorithm. See Appendix II for the detailed proof. ■

So far we have taken a deterministic model with a static population of users, and stability here means global asymptotic convergence. Given the dynamic nature of DATE, it is natural to wonder whether it would also behave well with stochastic variations in traffic. Consider sessions (equivalent to logical "sources") arriving according to a Poisson process with exponentially-distributed file sizes. A session leaves the network after it finishes transmitting a file. The service rates are determined by the solution of DATE algorithm. Note that sessions may arrive and depart even before the DATE algorithm converges, i.e., we do not assume time-scale separation between algorithm convergence and stochastic arrivals. The key question becomes one on the *stochastic stability* of DATE: whether the number of active sessions and the sizes of queues in the network remain finite for DATE in such dynamic environment. The answer is positive, as summarized in the following theorem, whose proof is to be found in [28].

Theorem 4: The DATE algorithm is stochastically stable if the average arrival load on each link is smaller than its capacity, i.e., the stochastic stability region of DATE is the largest possible one: the interior of the feasible region of problem (10).

Outline of the Proof: The key idea is to show that dual variables as scaled versions of queue lengths, and then show that the DATE algorithm follows as a special case of dual-based algorithms for generalized Network Utility Maximization whose stochastic stability has been recently established. ■

C. Simulation

Following the experimental set-up in section III, we use the Abilene topology in Figure 3(a). In all cases, the capacity distribution is a truncated Gaussian distribution with an average value of 100 and standard deviation is 10 unless otherwise specified. We say that convergence has occured when the aggregate utility is within 0.001 of the global optimum of (10).

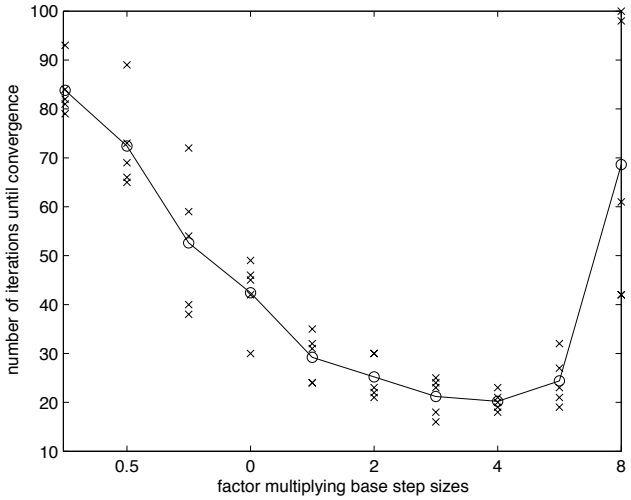
The graphs in Figure 9 illustrate both the rates and the aggregate utility converges within 30 iterations. In addition, the aggregate utility in Figure 9(b) follows an increasing concave trajectory, converging close to the optimum in less than 10 iterations. While the graphs in Figure 9 are for one particular initial condition, we have done simulations for a variety of initial conditions to verify that convergence time is independent of the initial conditions.

Figure 10(a) illustrates the rate of convergence is fairly insensitive to the step sizes within one order of magnitude. Further, Figure 10(b) shows the rate of convergence is also insensitive to the ratio between the step sizes within one order of magnitude. Practically, this implies it would be relatively easy for operators to find step sizes that work well for their networks. Figure 10(c) illustrates that the rate of convergence depends on the underlying capacity distribution. Luckily, most real backbone networks have relatively uniform capacity distributions, leading to very fast convergence.

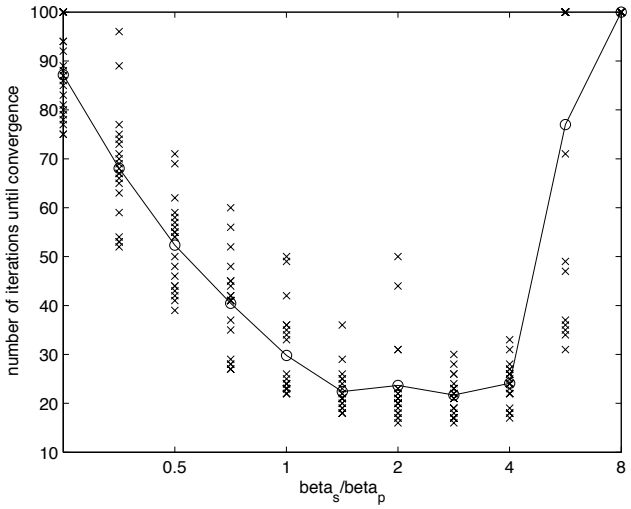
VII. Related Work

While there are cross-layer studies which bear some resemblance to our TE model, they share only the congestion control model or the routing model, but not both. A detailed analytic model of congestion control exists in [7], [8], [29], but they do not take the operator's cost function into account. In addition, [7], [8] find convergence to equilibrium is not guaranteed when congestion price used as link weights. In [30], the authors consider a related routing model based on centrally minimizing the maximum link utilization, but do not model congestion control analytically. There are a few papers *e.g.*, [31], [32], [33] that use the same routing model as in this paper, but the user's adaptation is modeled by overlay routing rather than TCP congestion control.

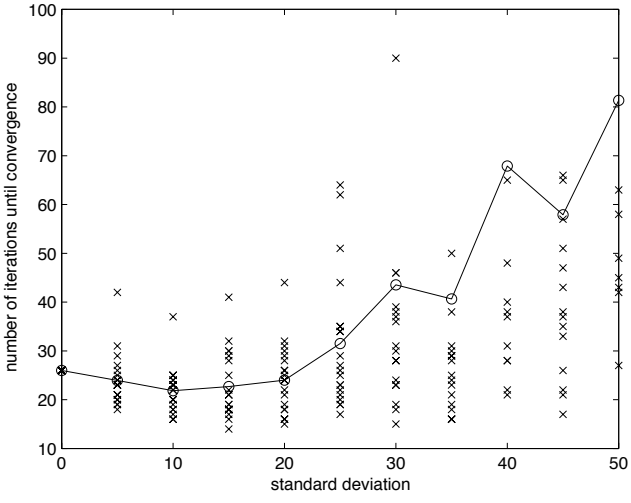
The DATE algorithm described in Section VI bears similarity to MATE [10], TeXCP [11] and REPLEX [14]. All four are traffic-engineering schemes that balance load across multiple paths using feedback from the links. The key difference is that the other schemes do not consider congestion control explicitly. In contrast, we start with a joint optimization



(a) Base step sizes: $\beta_s = 1 \times 10^{-5}$ and $\beta_p = 5 \times 10^{-5}$



(b) Base step size: $\beta_s = 1 \times 10^{-5}$



(c) $\beta_s = 4 \times 10^{-5}$ and $\beta_p = 2 \times 10^{-5}$ and $\beta_p = 5 \times 10^{-5}$

Fig. 10. Two plots of rate of convergence versus step size with standard deviation of 10 and one plot of rate of convergence versus standard deviation.

problem and derive an algorithm that maximizes aggregate user utility and minimizes network congestion. The solution approach is also different. MATE and TeXCP start with a heuristic and then prove convergence using control-theoretic tools, while REPLEX uses an algorithm derived from a game-theoretic approach. While DATE and TeXCP are quite different approaches, they both fix the usable paths and measure link utilization. TeXCP is a simpler algorithm, due to its simpler objective (to minimize the maximum link utilization); hence, TeXCP can be thought of as an incremental step towards deploying DATE. While REPLEX also covers multi-domain traffic engineering and has been more extensively simulated than DATE, it does not provide any guarantees regarding the efficiency of the Nash equilibrium.

There is a body of work on dynamic multipath routing algorithms from a control-theoretic approach studying stability through Lyapunov functions, *e.g.*, [12], [13], [34], [35], [36]. The seminal work in [3] considers the effect of multipath routing but does not optimize over routing decision. Both [13] and [12] consider dynamic multi-path routing with utility maximization as the model. In [12], they choose (6) as the joint optimization problem, hence their objective function does not take into account the operator's needs explicitly and is different from that in our new design. In [13], log utility is used whereas we allow general concave utility functions. The link cost model in [13] and [35] is based on link load, whereas our link cost is based on link utilization (as motivated by the practice of network operators), and does not explicitly consider the capacity constraints. Another difference is that [13] and [35] use the solution approach of differential equations based on general price feedback, while we use dual decomposition to derive DATE. While both approaches have similar overhead in terms of message passing and computation overhead, the model in [13] and [35] lets users determine their price rather than their throughput, which may be harder to implement in practice given ISPs tend to charge customers for their traffic on a much longer timescale, such as weeks or months. In [36], load balancing is done at two timescales, while it happens at a single timescale in DATE.

VIII. Conclusion

We studied the multi-layer interaction of congestion control and routing under two models. Congestion control by TCP and route adaptation by traffic engineering both try to make efficient use of link bandwidth to improve network performance for end users. In today's IP networks, however, these two mechanisms operate independently, though they are coupled because they both adapt to and also alleviate network congestion.

In the first half of this paper, we first find through simulation that TCP and traffic engineering work effectively together to reach a stable equilibrium that maximizes aggregate user utility under most network configurations. Analytic study then proves that a modification to the operator's cost function indeed leads to a provably stable and optimal system, but at the expense of robustness. This highlights the potential tension between performance and non-performance metrics.

Turning from analysis to design in the second half of the paper, we have defined an optimization problem where the objective includes end-user utilities and the network operator

cost function. A distributed solution (DATE) to this problem balances the tension between robustness and optimality in two ways. First, by incorporating the operator's cost function into the objective, DATE protects the network from short traffic bursts. Second, by finding a distributed solution, the algorithm can react to traffic shifts on a smaller timescale. Stability (both convergence in the deterministic fluid model and stochastic stability when session-level arrivals are considered), optimality, and robustness of DATE are proved and illustrated.

In our ongoing work, we will consider the effects of feedback delay using the control-theoretic tools that have successfully shown local asymptotic stability under delay for other network resource-allocation models [12], [13], [34]. We are in the process of evaluating DATE in the ns-2 simulator under realistic topologies and workloads. In particular, we are testing how DATE reacts to packet-level, session-level, and topology-level stochastics. In [37], we consider algorithms similar to DATE using alternative optimization decompositions. We take the best parts of different distributed solutions to construct an algorithm which can work with implicit feedback at the edge routers. This allows for the potential to extend traffic management to the multidomain scenario.

ACKNOWLEDGMENTS

We would like to thank Jim Kurose, Steven Low, Ao Tang and the anonymous reviewers for their insightful comments. This work has been supported in part by NSF grants CNS-0519880, CCF-0635034, CCF-0448012, Cisco grant GH072605, and DARPA seedling on Design for Manageability.

APPENDIX I
PROOF OF THEOREM 1

Proof: First we show that the joint routing and modified congestion control system (5,7) is equivalent to a successive, alternating optimization of (8) over $\mathbf{R}$ and then $\mathbf{x}$. Then we provide a sufficient condition to guarantee convergence to optimality. Finally the condition is examined for α-fair utilities and arctan utility.

Consider the unconstrained minimization of

$$g(\mathbf{x},\mathbf{R}) = -\sum_i U_i(x_i) + \sum_l f\left(\sum_i R_{li}x_i/c_l\right), \quad (11)$$

which is equivalent to (8). The two steps in the alternating optimization method of Gauss-Seidel algorithm [38] are as follows:

$$\begin{aligned}\mathbf{x}(t+1) &= \text{argmin}_{\mathbf{x}} -\sum_i U_i(x_i) + \sum_l f(\sum_i R_{li}(t)x_i/c_l)\\ \mathbf{R}(t+1) &= \text{argmin}_{\mathbf{R}} g(\mathbf{x}(t+1),\mathbf{R}(t))\\ &= \text{argmin}_{\mathbf{R}} \sum_l f(\sum_i R_{li}(t)x_i(t+1)/c_l).\end{aligned}$$

The minimization of $g(\mathbf{x},\mathbf{R})$ over $\mathbf{R}$ is clearly equivalent to (1).

So far we have constructed an optimization problem (minimization of $g(\mathbf{x},\mathbf{R})$) whose Gauss-Seidel solution algorithm is equivalent to the system model of joint routing and modified congestion control. Now we will examine the conditions for convergence of this Gauss-Seidel Algorithm. From [38], the Gauss-Seidel Algorithm will converge to the minimizer of g if g is bounded from below, differentiable, marginally strictly convex in $\mathbf{x}$ and $\mathbf{R}$, and jointly convex in $\mathbf{x}$ and $\mathbf{R}$.

The first three conditions are already satisfied through the constraints placed in the system model definition. Condition 1 is satisfied since $\mathbf{x} \succeq \mathbf{0}$, $\mathbf{R} \succeq \mathbf{0}$ by definition. Condition 2 is satisfied since U and f are differentiable, so is g. The third condition is satisfied since U is strictly concave in $\mathbf{x}$, and f is marginally strictly convex in $\mathbf{x}$ and $\mathbf{R}$. The last condition is not satisfied in general since the function $f(\sum_l R_{li}x_i/c_l)$ is not jointly convex in $\mathbf{R}$ and $\mathbf{x}$.

In order to satisfy the condition on joint convexity in $\mathbf{x}$ and $\mathbf{R}$, consider a log change of variable. Let $\tilde{x_i} = \log x_i, \tilde{R_{li}} = \log R_{li}$, then $R_{li}x_i = \exp(\tilde{R_{li}} + \tilde{x_i})$. With the change of variable, it can be readily verified that f is still jointly convex in $\tilde{x_i}$ and $\tilde{R_{li}}$, but the utility function may no longer be concave in $\tilde{\mathbf{x}}$. If the utility function is concave in $\tilde{\mathbf{x}}$, then g would be strictly convex in $\tilde{\mathbf{x}}$ since f is strictly convex in $\tilde{\mathbf{x}}$. Denote the new utility function (after the log change of variable) as $W_i(\tilde{x_i})$. A sufficient condition for convergence of the Gauss-Seidel algorithm is for W to be concave in $\tilde{\mathbf{x}}$. A simple derivation shows that such a condition reduces to the following simple bound on the curvature of the utility function: $U_i^{''}(x_i) \le -U_i'(x_i)/x_i$.

Now we specialize to the α-fairness model for U which covers TCP Reno (currently deployed) and several proposed variants. In this case, $W_\alpha(\tilde{\mathbf{x}})$ can be written as follows:

$$W_\alpha(\tilde{\mathbf{x}}) = \begin{cases} \tilde{\mathbf{x}}, & \alpha = 1 \\ (1-\alpha)^{-1}\exp(\mathbf{x})^{1-\alpha}, & \alpha \neq 1. \end{cases} \quad (12)$$

Examining $W^{''}(\tilde{\mathbf{x}})$ shows that $W(\tilde{\mathbf{x}})$ is concave for $\alpha \ge 1$.

Finally, TCP Tahoe is examined. Recall that $U(\mathbf{x}) = \arctan(\mathbf{x})$ for TCP Tahoe, and $W^{''}(\tilde{\mathbf{x}}) = \arctan(\mathbf{x})$. It follows that $W^{''}(\tilde{\mathbf{x}}) = (\exp(\tilde{\mathbf{x}}) - \exp(3\tilde{\mathbf{x}}))/(1+\exp(2\tilde{\mathbf{x}}))^2$ and W is concave. Therefore, convergence of (8) is guaranteed for TCP algorithms with $\alpha \ge 1$ and TCP Tahoe. ■

APPENDIX II
PROOF OF THEOREM 3

Proof: First we rewrite (10) as a convex problem by introducing new variable z_j^i:

$$\begin{aligned}\text{maximize} \quad & \textstyle\sum_i U_i(\mathbf{1}^T\mathbf{z}^i) - \sum_l f(y_l/c_l)\\ \text{subject to} \quad & \mathbf{y} \preceq \mathbf{c},\\ & y_l = \textstyle\sum_i \sum_j H_{lj}^i z_j^i, \ \ \forall l.\end{aligned} \quad (13)$$

In order to enable decoupling, we introduce a new variable $y_l = \sum_i \sum_j H_{lj}^i z_j^i, \ \forall l$, which also resulted in an additional constraint. Since (13) is a convex optimization problem satisfying Slater's condition, the duality gap is zero. Therefore, a distributed algorithm for (13) can be derived through the Lagrange dual problem. First we form the following Lagrangian:

$$\begin{aligned}L(\mathbf{z},\mathbf{y},\mathbf{p},\mathbf{s}) = \ & \textstyle\sum_i U_i(\mathbf{1}^T z^i) - \sum_l f(y_l/c_l)\\ & + \textstyle\sum_l p_l(c_l - y_l)\\ & + \textstyle\sum_l s_l(y_l - \sum_i \sum_j H_{lj}^i z_j^i).\end{aligned}$$

where $p_l \ge 0$ is the Lagrange multiplier associated with the capacity constraint on link l, and s_l is the Lagrange multiplier

associated with the equality constraint on link l. Additivity of total utility and linearity of flow constraints lead to a Lagrangian dual decomposition into two sub-problems:

1) Per Source i:

$$\text{argmax}_{z_j^i} U_i(\mathbf{1}^T \mathbf{z}^i) - z_j^i \sum_l s_l H_{lj}^i.$$

2) Per Link l:

$$\text{argmin}_{y_l} f(y_l/c_l) - (s_l + p_l) y_l.$$

The Lagrangian dual function $h(\mathbf{p}, \mathbf{s})$ is defined as the maximized $L(\mathbf{z}, \mathbf{y}, \mathbf{p}, \mathbf{s})$ over $\mathbf{z}$ and $\mathbf{y}$ for given p and s. Each source can compute an optimizer $\mathbf{z}^{i*}(\mathbf{s})$ and each link can compute an optimizer $y_l^*(p_l, s_l)$. The Lagrange dual problem of (13) is:

$$\begin{array}{ll} \text{minimize} & h(\mathbf{p}, \mathbf{s}) = L(\mathbf{z}^*(\mathbf{s}), \mathbf{y}^*(\mathbf{p}, \mathbf{s}), \mathbf{p}, \mathbf{s}) \\ \text{subject to} & \mathbf{p} \succeq \mathbf{0}, \end{array} \tag{14}$$

where $(\mathbf{p}, \mathbf{s})$ are the dual variables. Note that (14) is a convex minimization. Since $h(\mathbf{p}, \mathbf{s})$ may be non-differentiable, an iterative subgradient method can be used to update the dual variables $(\mathbf{p}, \mathbf{s})$ to solve (14):

1) Consistency Price Update:

$$p_l(t+1) = [p_l(t) - \beta_p(t)(c_l - y_l^*(t))]^+,$$

$\beta_p(t)$ represent the consistency price step size.

2) Congestion Price Update:

$$s_l(t+1) = s_l(t) - \beta_s(t) \left(y_l^*(t) - \sum_i \sum_j H_{lj}^i z_j^{*i} \right),$$

$\beta_s(t)$ represent the congestion price step size.

This is precisely the distributed algorithm described in SectionVI.A. Certain choices of step sizes, such as $\beta(t) = \beta_1/t, \beta_s(t) = \beta_2/t$ where $\beta_1 > 0, \beta_2 > 0$, guarantee that this algorithm will converge to the joint optimum [24]. In this case, the convergent point is a globally optimal $(\mathbf{R}, \mathbf{x})$ to problem (10) since we have shown that the problem can be written as convex optimization. ■

REFERENCES

[1] B. Fortz and M. Thorup, "Optimizing OSPF weights in a changing world," *IEEE J. on Selected Areas in Communications*, vol. 20, no. 4, pp. 756–767, May 2002.

[2] J. Rexford, "Route optimization in IP networks," in *Handbook of Optimization in Telecommunications*. Springer Science + Business Media, February 2006.

[3] F. P. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: Shadow prices, proportional fairness and stability," *J. of Operational Research Society*, vol. 49, no. 3, pp. 237–252, March 1998.

[4] S. H. Low, L. Peterson, and L. Wang, "Understanding Vegas: A duality model," *J. of the ACM*, vol. 49, no. 2, pp. 207–235, March 2002.

[5] S. H. Low, "A duality model of TCP and queue management algorithms," *IEEE/ACM Trans. Networking*, vol. 11, no. 4, pp. 525–536, August 2003.

[6] R. Srikant, *The Mathematics of Internet Congestion Control*. Birkhauser, 2004.

[7] J. Wang, L. Li, S. H. Low, and J. C. Doyle, "Cross-layer optimization in TCP/IP networks," *IEEE/ACM Trans. Networking*, vol. 13, no. 3, pp. 582–595, June 2005.

[8] J. He, M. Chiang, and J. Rexford, "TCP/IP Interaction Based on Congestion Price: Stability and Optimality," in *Proc. IEEE International Conference on Communications*, June 2006.

[9] M. Chiang, S. H. Low, R. A. Calderbank, and J. C. Doyle, "Layering as optimization decomposition," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 252–312, January.

[10] A. Elwalid, C. Jin, S. H. Low, and I. Widjaja, "MATE: MPLS Adaptive Traffic Engineering," in *Proc. IEEE INFOCOM*, April 2001.

[11] S. Kandula, D. Katabi, B. Davie, and A. Charny, "Walking the Tightrope: Responsive Yet Stable Traffic Engineering," in *Proc. ACM SIGCOMM*, August 2005.

[12] X. Lin and N. B. Shroff, "Utility Maximization for Communication Networks with Multi-path Routing," *IEEE Trans. Automatic Control*, vol. 51, no. 5, May 2006.

[13] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley, "Multi-Path TCP: A Joint Congestion Control and Routing Scheme to Exploit Path Diversity on the Internet," *IEEE/ACM Trans. Networking*, vol. 14, no. 6, December 2006.

[14] S. Fischer, N. Kammenhuber, and A. Feldmann, "REPLEX — Dynamic Traffic Engineering Based on Wardrop Routing Policies," in *Proc. CoNEXT*, December 2006.

[15] J. Moy, "OSPF Version 2," RFC 2328, April 1998.

[16] R. Callon, "Use of OSI IS–IS for Routing in TCP/IP and Dual Environments," RFC 1195, December 1990.

[17] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," RFC 3031, January 2001.

[18] S. Chen and K. Nahrstedt, "An overview of quality-of-service routing for the next generation high-speed networks: Problems and solutions," *IEEE Network Magazine*, 1998.

[19] F. Kuipers, T. Korkmaz, M. Krunz, and P. V. Mieghem, "Overview of constraint-based path selection algorithms for QoS routing," *IEEE Communication Magazine*, pp. 50–55, December 2002.

[20] S. Balon, F. Skive, and G. Leduc, "How well do traffic engineering objective functions meet TE requirements?" in *Proc. IFIP Networking*, May 2006.

[21] J. Mo and J. C. Walrand, "Fair End-to-end Window-based Congestion Control," *IEEE/ACM Trans. Networking*, vol. 8, no. 5, pp. 556–567, October 2000.

[22] MOSEK Optimization Software, http://www.mosek.com/.

[23] Abilene Backbone, http://abilene.internet2.edu/.

[24] D. P. Bersekas, *Nonlinear Programming*, 2nd ed. Athena Scientific, 1999.

[25] D. Bertsekas, "Dynamic behavior of shortest-path routing algorithms for communication networks," *IEEE Trans. Automatic Control*, pp. 60–74, February 1982.

[26] J. M. McQuillan and D. C. Walden, "The ARPA network design decision," *Computer Networks*, vol. 1, no. 5, pp. 243–289, August 1977.

[27] Z. Wang and J. Crowcroft, "Analysis of shortest-path routing algorithm in dynamic network environment," *ACM SIGCOMM Computer Communication Review*, vol. 22, no. 2, pp. 63–71, April 1992.

[28] J. Liu, M. Chiang, and V. Poor, "Stochastic Stability of General Optimization-Based Network Resource Allocation," preprint.

[29] H. Che, W. Su, C. Lagoa, K. Xu, C. Liu, and Y. Cui, "An integrated, distributed traffic control strategy for the future Internet," in *Proc. SIGCOMM Workshop on Internet Network Management*, September 2006, pp. 17–22.

[30] E. J. Anderson and T. E. Anderson, "On the stability of adaptive routing in the presence of congestion control," in *Proc. IEEE INFOCOM*, April 2003.

[31] Y. Liu, H. Zhang, W. Gong, and D. Towsley, "On the Interaction Between Overlay Routing and Traffic Engineering," in *Proc. IEEE INFOCOM*, April 2005.

[32] L. Qiu, Y. R. Yang, Y. Zhang, and S. Shenker, "On Selfish Routing in Internet-Like Environments," in *Proc. ACM SIGCOMM*, August 2003.

[33] R. Keralapura, C.-N. Chuah, N. Taft, and G. Iannaccone, "Can coexisting overlays inadvertently step on each other?" in *Proc. International Conference on Network Protocols*, November 2005.

[34] F. Kelly and T. Voice, "Stability of end-to-end algorithms for joint routing and rate control," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 2, pp. 5–12, April 2005.

[35] R. J. Gibben and F. Kelly, "On packet marking at priority queues," *IEEE Trans. Automatic Control*, vol. 47, no. 12, pp. 1016–1020, December 2002.

[36] P. Key, L. Massoulie, and D. Towsley, "Combining multipath routing and congestion control for robustness," in *Conference on Information Sciences and Systems*, March 2006.

[37] J. He, M. Bresler, M. Chiang, and J. Rexford, "Rethinking Traffic Management: From Multiple Decompositions to A Practical Protocol," 2007, tech. Report TR-774-07.

[38] D. P. Bersekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, 2nd ed. Athena Scientific, 1997.

Jiayue He (S'05) received her B.A.Sc. (Hon.) in Engineering Science from University of Toronto in 2004. She is currently working towards her Ph. D. degree at Princeton University. She interned at Thomson Paris Research Labs in summer 2006. Her PhD is partially funded by the Gordon Wu Fellowship at Princeton University and the graduate fellowship from National Science and Engineering Research Council of Canada. She is an Anita Borg Google Scholarship finalist, and received IEEE Globecom Best Student Paper Award in 2006.

Ma'ayan Bresler received her B.A. (High Hon.) in Physics with a certificate in the Program for Applied and Computational Mathematics from Princeton University in 2006. She received the Kusaka Memorial Prize in physics (2006) and the Lucent Prize in physics (2003). Her senior independent work was conducted with Mung Chiang and she also worked as a member of his group in summer 2006.

Mung Chiang (S'00, M'03) is an Assistant Professor of Electrical Engineering, and an affiliated faculty of the Program in Applied and Computational Mathematics at Princeton University. He received the B.S. (Hon.) degree in electrical engineering and in mathematics, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, CA, in 1999, 2000, and 2003, respectively. He conducts research in the areas of optimization of communication systems, theoretical foundations of network architectures, broadband access networks, and stochastic theory of communications and networking. He has been awarded a Hertz Foundation Fellowship, and received the Stanford University School of Engineering Terman Award, the SBC Communications New Technology Introduction contribution award, the NSF CAREER Award, and the Princeton University Howard B. Wentz Junior Faculty Award. He is a Co-Editor of Springer book series on Optimization and Control of Communication Systems, the Lead Guest Editor of the IEEE JSAC special issue on Nonlinear Optimization of Communication Systems, a Guest Editor of the IEEE Tran. Inform. Theory and IEEE/ACM Tran. Networking joint special issue on Networking and Information Theory, and the Program Co-Chair of the 38th Conf. Inform. Science and Systems.

Jennifer Rexford (S'89, M'96, SM'01) Jennifer Rexford is a Professor in the Computer Science department at Princeton University. From 1996-2004, she was a member of the Network Management and Performance department at AT &T Labs–Research. Jennifer is co-author of *Web Protocols and Practice* (Addison-Wesley, May 2001). Jennifer serves as the chair of ACM SIGCOMM and a member the CRA Board of Directors. She received her BSE degree in electrical engineering from Princeton University in 1991, and her MSE and PhD degrees in computer science and electrical engineering from the University of Michigan in 1993 and 1996, respectively. She was the winner of ACM's Grace Murray Hopper Award for outstanding young computer professional of the year for 2004.