Oblivious routing of highly variable traffic in service overlays and IP backbones

Citation: Kodialam, M. et al. "Oblivious Routing of Highly Variable Traffic in Service Overlays and IP Backbones." Networking, IEEE/ACM Transactions on 17.2 (2009): 459-472. © 2009 IEEE

As Published: http://dx.doi.org/10.1109/TNET.2008.927257

Publisher: Institute of Electrical and Electronics Engineers

Persistent URL: http://hdl.handle.net/1721.1/52466

Version: Final published version: final published article, as it appeared in a journal, conference proceedings, or other formally published context

Oblivious Routing of Highly Variable Traffic in Service Overlays and IP Backbones

Murali Kodialam, *Associate Member, IEEE*, T. V. Lakshman, *Fellow, IEEE*, James B. Orlin, and Sudipta Sengupta, *Senior Member, IEEE*

***Abstract*—The emergence of new applications on the Internet like voice-over-IP, peer-to-peer, and video-on-demand has created highly dynamic and changing traffic patterns. In order to route such traffic with quality-of-service (QoS) guarantees *without requiring detection of traffic changes in real-time or reconfiguring the network in response to it*, a routing and bandwidth allocation scheme has been recently proposed that allows preconfiguration of the network such that all traffic patterns permissible within the network's natural ingress–egress capacity constraints can be handled in a capacity efficient manner. The scheme routes traffic in two phases. In the first phase, incoming traffic is sent from the source to a set of intermediate nodes and then, in the second phase, from the intermediate nodes to the final destination. The traffic in the first phase is distributed to the intermediate nodes in predetermined proportions that depend on the intermediate nodes. In this paper, we develop linear programming formulations and a fast combinatorial algorithm for routing under the scheme so as to maximize throughput (or, minimize maximum link utilization). We compare the throughput performance of the scheme with that of the optimal scheme among the class of all schemes that are allowed to even make the routing dependent on the traffic matrix. For our evaluations, we use actual Internet Service Provider topologies collected for the Rocketfuel project. We also bring out the versatility of the scheme in not only handling widely fluctuating traffic but also accommodating applicability to several widely differing networking scenarios, including i) economical Virtual Private Networks (VPNs); ii) supporting indirection in specialized service overlay models like Internet Indirection Infrastructure (i3); iii) adding QoS guarantees to services that require routing through a network-based middlebox; and iv) reducing IP layer transit traffic and handling extreme traffic variability in IP-over-optical networks without dynamic reconfiguration of the optical layer. The two desirable properties of supporting indirection in specialized service overlay models and static optical layer provisioning in IP-over-optical networks are not present in other approaches for routing variable traffic, such as direct source–destination routing along fixed paths.**

***Index Terms*—Hose traffic model, IP/MPLS, IP-over-optical, oblivious routing, service overlays, two-phase routing, variable traffic, valiant load balancing.**

I. Introduction

As THE Internet continues to grow in size and complexity, it becomes increasingly difficult to predict future traffic patterns. Many emerging applications for the Internet are characterized by highly variable traffic behavior over time. Classical approaches to network design and planning rely on a model in which a single traffic matrix is estimated. When actual traffic does not conform to such assumptions (as is often the case), desired quality-of-service (QoS) cannot be guaranteed due to network congestion. Development of routing infrastructures that optimize network resources while accommodating extreme traffic unpredictability in a robust and efficient manner will be one of the defining themes in the next phase of expansion of the Internet.

In order to meet this requirement of robust and efficient network routing in a highly dynamic and changing traffic environment, a routing and bandwidth allocation scheme has been recently proposed in [9] and [25] that allows preconfiguration of the network such that all traffic patterns permissible within the network's natural ingress–egress capacity constraints can be handled without network reconfiguration. Such preconfiguration simplifies network operation by avoiding the need to detect traffic changes in real-time and to reconfigure the network in response. The scheme routes traffic in two phases. In the first phase, incoming traffic is sent from the source to a set of intermediate nodes and then, in the second phase, from the intermediate nodes to the final destination. The traffic in the first phase is distributed to the intermediate nodes in predetermined proportions that depend on the intermediate nodes, as proposed in [9]. Throughout this paper, we will refer to this scheme as *two-phase routing*.

In order to fully comprehend the motivation behind the development of such a scheme, it is important to understand, from an Internet Service Provider's (ISP) perspective, the difficulty of deploying and operating a more dynamic architecture that requires the measurement of possibly changing traffic in real-time as well as reconfiguring the network in response to such changes in order to provide QoS guarantees. We address these aspects in Sections II-A and IV.

The two-phase routing scheme is versatile not only in its ability to handle widely fluctuating traffic but also in its applicability to several widely differing networking scenarios. We illustrate this through example applications of the routing scheme to i) economical Virtual Private Networks (VPNs); ii) providing Internet Indirection Infrastructure (i3) [22] like functionality with QoS guarantees in a network; iii) adding QoS guarantees to services that require routing through a network-based middlebox; and iv) reducing IP layer transit traffic and handling extreme traffic variability in IP-over-optical networks without dynamic reconfiguration of the optical layer.

A unique aspect of the i3 application arising from its indirection property is that unlike traditional networks, the final destination of a packet is not known at the network ingress. Hence,

Manuscript received December 18, 2006; revised August 26, 2007; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor Z.-L. Zhang. First published July 25, 2008; current version published April 15, 2009.

M. Kodialam and T. V. Lakshman are with Bell Laboratories, Alcatel-Lucent, Murray Hill, NJ 07974 USA.

J. B. Orlin is with the Massachusetts Institute of Technology, Cambridge, MA 02139 USA.

S. Sengupta is with Microsoft Research, Redmond, WA 98052 USA (e-mail: sudipta@lcs.mit.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2008.927257

methods that need pre-provisioned paths to be set up between a network's ingress and egress nodes for providing bandwidth guarantees are not usable. For the IP-over-optical network application, it is important that both paths and their associated bandwidths do not change with shifts in traffic. The scheme is well-suited to both these applications unlike existing routing methodologies.

We develop linear programming formulations and a fast combinatorial algorithm for routing under the scheme so as to maximize throughput (or, minimize maximum link utilization). We compare the throughput performance of two-phase routing with that of the optimal scheme among the class of all schemes that are allowed to make the routing dependent on the traffic matrix. For our evaluations, we use actual ISP network topologies collected for the Rocketfuel project [21].

The combinatorial algorithm developed is a Fully Polynomial Time Approximation Scheme (FPTAS). An FPTAS is an algorithm that finds a solution with objective function value within $(1+\epsilon)$-factor of the optimal solution and runs in time that is a polynomial function of the input parameters and $1/\epsilon$. The input parameters in our problem are the number of nodes n and links m in the network, and the size (number of bits) of the input numbers (link capacities and node ingress–egress capacities). The value of ϵ can be chosen to provide the desired degree of optimality for the solution.

Throughput (which is the reciprocal of maximum link utilization) is an important but not the only optimization metric for network routing. For example, network capacity minimization has been considered in the context of two-phase routing in [9]. We focus on network throughput in this paper because it is one of the most common metrics used in the literature, it is used in capacity planning decisions by ISPs, it is directly related to other metrics like link congestion, and is useful for multi-period traffic planning when the traffic patterns scale (roughly) uniformly over time. When considering feasibility of a traffic matrix on (various what-if) capacitated network deployment scenarios, throughput is probably the most suitable metric to consider (feasibility is indicated by a throughput greater than or equal to 1).

The paper is structured as follows. In Section II, we discuss some aspects of the inherent difficulty in measuring traffic and introduce the traffic variation model. In Section III, we describe some application scenarios and their requirements. In Section IV, we argue why two of these requirements are not met by existing routing methodologies, thus making the case for two-phase routing. In Section V, we briefly discuss the two-phase routing scheme so as to provide context for this paper. We return to the application scenarios in Section VI and explain how two-phase routing meets the requirements outlined in Section III. Section VII introduces the throughput maximization problem for two-phase routing and provides linear programming formulations. In Section VIII, we develop a fast combinatorial algorithm for the problem. Performance evaluation of two-phase routing is presented in Section IX. Finally, we conclude in Section X. Proofs of theorems establishing the performance guarantees and running times of the fast combinatorial algorithm are presented in Appendix A. We briefly describe some notation before moving on to the next section.

A. Notation

We assume that we are given a network $G = (N, E)$ with node set N and (directed) edge set E where each node in the network can be a source or destination of traffic. Let $|N| = n$ and $|E| = m$. The nodes in N are labeled $\{1, 2, \ldots, n\}$. The sets of incoming and outgoing edges at node i are denoted by $E^-(i)$ and $E^+(i)$, respectively. We let (i, j) represent a directed link in the network from node i to node j. To simplify the notation, we will also refer to a link by e instead of (i, j). The capacity of link (i, j) will be denoted by u_{ij}. The *utilization* of a link is defined as the traffic on the link divided by its capacity.

II. Traffic Measurement and Variability

In a utopian network deployment scenario where complete traffic information is known and does not change over time, we can optimize the routing for that single traffic matrix—a large volume of research has addressed this problem. The most important innovation of the two-phase routing scheme is the handling of traffic variability in a capacity efficient manner through static preconfiguration of the network and without requiring either (i) measurement of traffic in real-time or (ii) reconfiguration of the network in response to changes in it. We address the difficulties associated with (i) in this section and then introduce the traffic variation model. The difficulties associated with (ii) for IP-over-optical networks are addressed in Section IV.

A. Difficulties in Measuring Traffic

Network traffic is not only hard to measure in real-time but even harder to predict based on past measurements. Direct measurement methods do not scale with network size as the number of entries in a traffic matrix is quadratic in the number of nodes. Moreover, such direct real-time monitoring methods lead to unacceptable degradation in router performance. In reality, only aggregate link traffic counts are available for traffic matrix estimation. SNMP (Simple Network Management Protocol) provides this data via incoming and outgoing byte counts computed per link every 5 minutes. To estimate the traffic matrix from such link traffic measurements, the best techniques today give errors of 20% or more [16].

The emergence of new applications on the Internet like voice-over-IP, peer-to-peer, and video-on-demand has reduced the time-scales at which traffic changes dynamically, making it impossible to extrapolate past traffic patterns to the future. Currently, ISPs handle such unpredictability in network traffic by gross over-provisioning of capacity. This has led to ISP networks being under-utilized to as low as 20% [16].

B. Traffic Variation Model

We consider a traffic variation model where the total amount of traffic that enters (leaves) an ingress (egress) node in the network is bounded by the total capacity of all external ingress links at that node. This is known as the *hose model* and was proposed by Fingerhut *et al.* [7] and subsequently used by Duffield *et al.* [6] as a method for specifying the bandwidth requirements of a Virtual Private Network (VPN). Note that the hose model naturally accommodates the network's ingress–egress capacity constraints.

We denote the upper bounds on the total amount of traffic entering and leaving the network at node i by R_i and C_i, respectively. The point-to-point matrix for the traffic in the network is thus constrained by these ingress–egress link capacity bounds. These constraints are the only known aspects of the traffic to be carried by the network, and knowing these is equivalent to knowing the row and column sum bounds on the traffic matrix. That is, any allowable traffic matrix $T = [t_{ij}]$ for the network must obey

$$\sum_{j \in N, j \neq i} t_{ij} \leq R_i, \quad \sum_{j \in N, j \neq i} t_{ji} \leq C_i \forall i \in N.$$

For given R_i and C_i values, denote the set of all such matrices that are partially specified by their row and column sums by $\mathcal{T}(\vec{R}, \vec{C})$, that is,

$$\mathcal{T}(\vec{R}, \vec{C}) = \left\{ [t_{ij}] \middle| \sum_{j \neq i} t_{ij} \leq R_i \text{ and } \sum_{j \neq i} t_{ji} \leq C_i \forall i \right\}.$$

We will use $\lambda \cdot \mathcal{T}(\vec{R}, \vec{C})$ to denote the set of all traffic matrices in $\mathcal{T}(\vec{R}, \vec{C})$ with their entries multiplied by λ.

Note that the traffic distribution T could be any matrix in $\mathcal{T}(\vec{R}, \vec{C})$ and could change over time. Two-phase routing provides a routing architecture that does not make any assumptions about T apart from the fact that it is partially specified by row and column sum bounds and can provide QoS guarantees for routing all matrices in $\mathcal{T}(\vec{R}, \vec{C})$ without requiring any detection of changes in traffic patterns or dynamic network reconfiguration in response to it.

III. Motivating Networking Applications

We discuss some motivating networking architectures and applications that need to handle traffic variation and identify the requirements of a suitable routing scheme for each scenario. In the next section, we then argue why such requirements are not met by existing routing methodologies.

A. IP Backbones

Core (long-haul) networks of ISPs form the backbone of the Internet and span vast geographical areas (countries and continents). Each node in such a network, also called a Point-of-Presence (PoP), connects an access network (or, regional/metro network) to the core network. Internet backbones are often deployed by interconnecting routers over a switched optical backbone, also called an *IP-over-optical network*. Because a router line card is typically 3–4 times more expensive than an optical switch card, an IP-over-optical network architecture reduces network cost by keeping traffic mostly in the optical layer [19]. By removing transit traffic from the routers to the optical switches, the requirement to upgrade router PoP configurations with increasing traffic is minimized (since optical switches are more scalable with increasing port count than routers). Also, since optical switches are known to be much more reliable compared to routers [15], this makes the architecture more robust and reliable.

Routing in IP-over-optical networks needs to make a compromise between keeping traffic at the optical layer (for the above reasons) and using intermediate routers for packet grooming in order to achieve efficient statistical multiplexing of data traffic. In addition, the routing must be able to handle traffic variability. The (current) traffic matrix is not only difficult to estimate but changes in the same may not be detectable in real time. Moreover, dynamic changes in routing in the network may be difficult or prohibitively expensive from a network operations perspective. In spite of the continuing research on IP-Optical integration, network deployments are far away from utilizing the optical control plane to provide bandwidth provisioning in real-time to the IP layer. These translate to the following requirements on the routing methodology:

- IP traffic must be routed "mostly" at the optical layer from source to destination routers. Intermediate IP layer transit may be required for grooming purposes.
- The optical layer (circuits and their bandwidth) must be statically provisioned a priori to provide bandwidth guarantees for end-to-end IP traffic. Routing at the IP layer cannot also be adaptive to traffic changes.
- Bandwidth guarantees must be provided for routing all traffic matrices.

B. Specialized Service Overlays

The Internet Indirection Infrastructure (i3) was proposed in [22] to ease the deployment of services—like mobility, multicast and anycast—on the Internet. i3 provides a rendezvous-based communication abstraction through indirection—sources send packets to a logical identifier, and receivers express interest in packets sent to an identifier. The rendezvous points are provided by i3 servers that forward packets to all receivers that express interest in a particular identifier. The communication between senders and receivers is through these rendezvous points over an overlay network. The i3 infrastructure does not store packets but only forwards them. It is important to note that i3 provides only a best-effort service like today's Internet—it neither implements reliability nor guarantees ordered delivery on top of IP.

Two-phase routing can support indirection and provide *QoS guarantees* for variable traffic in specialized service overlays like i3. This is discussed in Section VI-B. Three important requirements in this context are:

- The routing from the source node(s) to the rendezvous points cannot depend on the final destination(s) of the packet, since this is unknown at the source.
- The traffic from the source nodes to the rendezvous points and from the latter to the destination nodes must be routed along bandwidth-guaranteed paths.
- These paths cannot be re-routed in response to changes in traffic patterns, and must have sufficient bandwidth to handle all possible traffic patterns subject to network ingress–egress constraints.

C. Middlebox Routing

Intermediate network elements (so called middleboxes), such as firewalls and transparent caches, are now commonplace. They provide important services like caching, load-balancing, and content filtering (for network security). To be effective, the services provided by such middleboxes are required to be comprehensive in the sense that every packet routed through the network must pass through at least one middlebox providing the service. In order to support a middlebox routing architecture, the

routing scheme needs to not only provide bandwidth guarantees for variable traffic but also handle the additional constraint that all network traffic must pass through at least one intermediate network element node. Two-phase routing can naturally accommodate such an architecture in ISP networks. This is discussed in Section VI-C.

D. Other Scenarios of Interest

Another example application where the traffic matrix is unknown is the provisioning of network-based VPN services [3] to enterprise customers. VPNs typically provide network connectivity among different sites of an enterprise. The traffic distribution between the sites is not known a priori—it may also change depending on time-of-day, day-of-week, special activities, etc. The enterprise customer specifies to the ISP only the total traffic volume and the peak rate out of a given site (e.g., if a site is connected to the ISP through a T1 link, this peak rate is about 1.5 Mb/s). It is the ISP's task to transport all of the offered VPN traffic to the network and carry the traffic in accordance with the bandwidth guarantees provided in the Service Level Agreement (SLA). The traffic originating from or destined to a VPN node is limited only by the aggregate bandwidth connection of that node to the VPN.

Networks for grid computing also need to handle highly variable traffic patterns. In grid computing, a complex computational task is partitioned amongst different computing nodes that can be geographically distributed and are connected by a network. The communication patterns amongst grid computing nodes are highly unpredictable and also can require high burst rates. Since the traffic matrix is not known, one option is to dynamically reserve capacity over an underlying network but this approach will be too slow for grid computing applications.

IV. Related Routing Methodologies

We briefly review related work on routing with traffic variability and point out why such existing methods cannot meet the requirements outlined above for the various application scenarios.

Direct routing from source to destination (instead of in two phases) along *fixed* paths for the hose traffic model has been considered by Duffield *et al.* [6] and Kumar *et al.* [14]. In related work, Azar *et al.* [2] consider direct source–destination routing along fixed paths and show how to compute *relative guarantees* for routing an arbitrary traffic matrix with respect to the best routing for that matrix. However, they do not provide *absolute bandwidth guarantees* for routing variable traffic under the hose model.

In both these approaches, *direct source–destination paths* are fixed a priori for routing the traffic between each source–destination pair. Thus, the source needs to *know the destination of a packet* for routing it, without which the source cannot determine the path along which the packet should be forwarded. In specialized service overlay models like i3, the *final destination of a packet is not known at the source*. Thus, any of the above approaches cannot be used for routing in service overlay networks.

Direct source–destination routing, when applied to IP-over-optical networks, routes packets from source to destination along direct paths in the optical layer. Note that

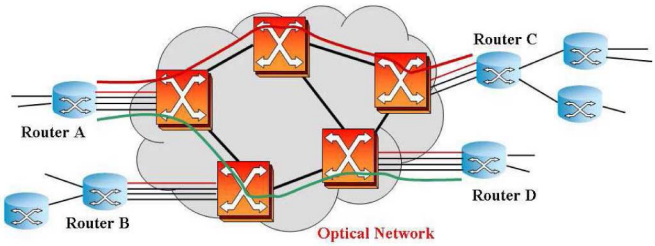

Fig. 1. Routing through direct optical layer circuits in IP-over-optical networks.

even though the paths are fixed a priori and do not depend on the traffic matrix, their *bandwidth requirements change* with variations in the traffic matrix. Thus, bandwidth needs to be deallocated from some paths and assigned to other paths as the traffic matrix changes. (Alternatively, paths between every source–destination pair can be provisioned a priori to handle the maximum traffic between them, but this leads to gross overprovisioning of capacity, since all source–destination pairs cannot simultaneously reach their peak traffic limit in the hose traffic model.) This necessitates *dynamic reconfiguration* of the provisioned optical layer circuits (i.e., change in bandwidth) in response to traffic variations, thus making direct source–destination routing unsuitable for IP-over-optical networks.

To illustrate this last point, consider the scenario in Fig. 1 for direct source–destination routing in IP-over-optical networks. Here, router A is connected to router C using 3 OC-48 connections and to Router D using 1 OC-12 connection, so as to meet the traffic demand from node A to nodes C and D of 7.5 Gb/s and 600 Mb/s, respectively. Suppose that at a later time, traffic from A to C decreases to 5 Gb/s, while traffic from A to D increases to 1200 Mb/s. Then, the optical layer must be reconfigured so as to delete one OC-48 connection between A and C and creating a new OC-12 connection between A and D. As such, the *requirement of static provisioning at the optical layer is not met*.

In contrast, two-phase routing has the following properties. They address both of the above issues.

- The source routes packets independent of their intended (or, unknown) destination, and
- Both the paths and their bandwidth are fixed a priori and do not need to be changed as traffic patterns change over time.

V. Overview of Two-Phase Routing

In this section, we give an overview of the two-phase routing scheme from [9]. As mentioned earlier, the scheme does not require the network to detect changes in the traffic distribution or reconfigure the network in response to it. The only assumption about the traffic is the limits imposed by the ingress–egress constraints at each node, as outlined in Section II-B.

As is indicative from the name, the routing scheme operates in two phases:

- **Phase 1**: A predetermined fraction α_j of the traffic entering the network at any node is distributed to every node j *independent of the final destination of the traffic*.
- **Phase 2**: As a result of the routing in Phase 1, each node receives traffic destined for different destinations that it routes to their respective destinations in this phase.

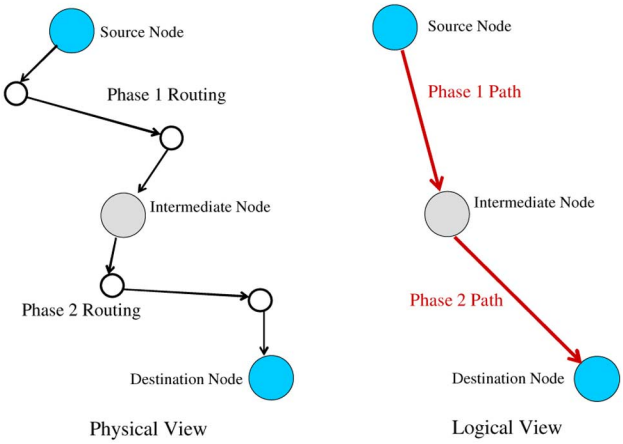

Fig. 2. Two-phase routing.

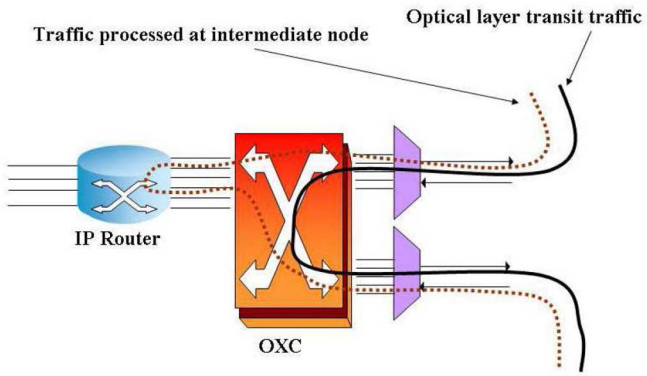

Fig. 3. Intermediate node packet processing for two-phase routing in IP-over-optical networks.

This is illustrated in Fig. 2. Note that the traffic split ratios $\alpha_1, \alpha_2, \ldots, \alpha_n$ in Phase 1 of the scheme are such that $\sum_{i=1}^{n} \alpha_i = 1$. A simple method of implementing this routing scheme in the network is to form *fixed bandwidth paths between the nodes*. In order to differentiate between the paths carrying Phase 1 and Phase 2 traffic, we will refer to them as Phase 1 and Phase 2 paths, respectively. The critical reason the two-phase routing strategy works is that the *bandwidth required for these tunnels depends on the ingress–egress capacities R_i, C_i and the traffic split ratios α_j but not on the (unknown) individual entries in the traffic matrix*.

Depending on the underlying routing architecture, the Phase 1 and Phase 2 paths can be implemented as IP tunnels, optical layer circuits, or Label Switched Paths in Multi-Protocol Label Switching (MPLS) [17]. The main requirement is to split traffic according to pre-determined ratios at each ingress router—this can be implemented using techniques standardized for multi-path routing in RFC 2991 [23].

We now derive the bandwidth requirement for the Phase 1 and Phase 2 paths. Consider a node i with maximum incoming traffic R_i. Node i sends $\alpha_j R_i$ amount of this traffic to node j during the first phase for each $j \in N$. Thus, the traffic demand from node i to node j as a result of Phase 1 routing is $\alpha_j R_i$. At the end of Phase 1, node i has received $\alpha_i R_k$ traffic from any other node k. Out of this, the traffic destined for node j is $\alpha_i t_{kj}$ since all traffic is initially split without regard to the final destination. The traffic that needs to be routed from node i to node j during Phase 2 is $\sum_{k \in N} \alpha_i t_{kj} \leq \alpha_i C_j$. Thus, the traffic demand from node i to node j as a result of Phase 2 routing is $\alpha_i C_j$.

Hence, the maximum demand from node i to node j as a result of routing in Phases 1 and 2 is $\alpha_j R_i + \alpha_i C_j$. Note that this does not depend on the matrix $T \in \mathcal{T}(\vec{R}, \vec{C})$. The scheme handles variability in traffic matrix $T \in \mathcal{T}(\vec{R}, \vec{C})$ by effectively routing the fixed matrix $D = [d_{ij}] = [\alpha_j R_i + \alpha_i C_j]$ that depends only on aggregate ingress–egress capacities and the traffic split ratios $\alpha_1, \alpha_2, \ldots, \alpha_n$, and not on the specific matrix $T \in \mathcal{T}(\vec{R}, \vec{C})$. This is what makes the routing scheme oblivious to changes in the traffic distribution.

An instance of the scheme requires specification of the traffic split ratios $\alpha_1, \alpha_2, \ldots, \alpha_n$ and routing of the Phase 1 and Phase 2 paths. Computation of the above so as to maximize network throughput is the main focus of this paper.

The scheme can be made resilient against link failures by protecting the first and second phase paths using pre-provisioned restoration mechanisms, such as local (link/span) restoration [13], K-route path restoration [13], and shared backup path restoration [18].

The traffic split ratios α_i can be generalized to depend on source and/or destination nodes of the traffic, as proposed in [9]. While this does not meet the indirection requirement of specialized service overlays like i3, it can potentially increase the throughput performance of the two-phase routing scheme for other application scenarios like IP-over-optical networks. We consider the problem of maximum throughput two-phase routing with generalized traffic split ratios in [12].

VI. Applications of Two-Phase Routing

We now return to the application scenarios described in Section III and discuss how our routing scheme can be applied to each scenario.

A. IP Backbones

Two-phase routing, as envisaged for IP-over-optical networks, establishes the fixed bandwidth Phase 1 and Phase 2 paths at the optical layer. Thus, the *optical layer is statically provisioned* and does not need to be reconfigured in response to traffic changes. IP packets are routed end-to-end with *IP layer processing at a single intermediate node only*. While in transit at the optical layer inside either Phase 1 or Phase 2 tunnels, packets do enter the router but appear as transit traffic at the Optical Cross-Connect (OXC) only. The IP layer packet processing at an intermediate node works as follows. The optical layer circuit is dropped at the IP router at the node (through OXC-to-router links), wherein the packets are multiplexed back to the OXC (through router-to-OXC links) to be routed through direct optical layer circuits to their final destinations. Fig. 3 illustrates optical layer transit traffic and intermediate node packet processing functionality at a node.

This architecture provides the desirable statistical multiplexing properties of packet switching for handling highly variable traffic *without significantly increasing the IP layer transit*. Compare this with the high levels of IP layer transit traffic in IP-over-WDM architecture where routers are directly connected to WDM systems and need to process packets at each hop. Protecting against intermediate node router failures for two-phase routing in IP-over-optical networks is considered in [11].

B. Specialized Service Overlays

Two-phase routing can be used to provide QoS guarantees for variable traffic and support indirection in intra-ISP deployments of specialized service overlays like i3. (Note that we are not considering Internet-wide deployment here.) The intermediate nodes in the two-phase routing scheme are ideal candidates for locating i3 servers. Because we are considering a network whose topology is known, two-phase routing can be used to not only pick the i3 server locations (intermediate nodes) but also traffic engineer paths for routing with bandwidth guarantees between sender and receiver through i3 server nodes. Because the two-phase routing scheme can route the Phase 1 and Phase 2 paths with protection, this can also provide network level reliability of the services provided.

The ingress–egress traffic constraints R_i, C_j in the two-phase routing scheme now apply to network nodes to which hosts attach for using the services provided. For example, the host could be a laptop and a node could be a corporate site or an ISP PoP. Mobility of the hosts manifest itself as changes in traffic originating from or destined to the network points of attachment (nodes), since mobile hosts will attach themselves to different nodes over time. The Phase 1 and Phase 2 paths of the specified bandwidth will provide bandwidth guarantees across all i3 applications described in [22], including mobility, multicast, and anycast. This is because the traffic arising from such applications obey, by default, the aggregate ingress–egress constraints at each node.

C. Middlebox Routing

Two-phase routing can naturally accommodate a middlebox routing architecture in ISP networks and also provide QoS guarantees for variable traffic. The intermediate nodes in two-phase routing are ideal locations for deploying middleboxes that provide functionalities like caching and content filtering. Because all traffic passes through one of the intermediate nodes in the scheme, the requirement of the middleware service to be comprehensive (in the sense that every packet routed through the network must be examined at least once) is also met. The routing can now provide end-to-end bandwidth guarantees for variable traffic patterns. Experiments on actual ISP topologies for maximum throughput two-phase routing in Section IX indicate that the number of intermediate nodes in two-phase routing is small compared to the total number of nodes in the network. Given that the deployment of services like content filtering are expensive (from a hardware perspective), a smaller number of intermediate nodes can lead to cost-effective deployment of such services.

VII. Linear Programming Formulations

Given a network with link capacities u_e and constraints R_i, C_j on the ingress–egress traffic, we consider the problem of two-phase routing so as to maximize throughput. The throughput is the maximum multiplier λ such that all matrices in $\lambda \cdot \mathcal{T}(\vec{R}, \vec{C})$ can be feasibly routed.

In this section, we describe linear programming (LP) formulations for the above problem. Note that for the case of equal split ratios, i.e., $\alpha_i = 1/n$ for all $i \in N$, the demand between nodes i and j is $(R_i + C_j)/n$, and the problem reduces to the maximum concurrent flow problem [20].

Suppose we relax the requirement that the traffic split ratios α_j sum to 1 in a feasible solution of the problem. Consider the sum $\lambda = \sum_{i \in N} \alpha_i$. The traffic split ratios can be divided by λ (normalized) so that they sum to 1, in which case all matrices in $\lambda \cdot \mathcal{T}(\vec{R}, \vec{C})$ can be feasibly routed. Thus, the appropriate measure of throughput is the quantity $\sum_{i \in N} \alpha_i$ *when the traffic split ratios are not constrained to sum to 1*.

A. Link Flow Based Formulation

We adopt the standard network flow terminology from [1]. Let x_e^{ij} denote the flow value on link e for routing $\alpha_j R_i + \alpha_i C_j$ amount of flow from source node i to destination node j. Then, the problem of two-phase routing so as to maximize throughput can be expressed as the following link indexed linear program:

$$\text{maximize} \sum_{i \in N} \alpha_i$$

subject to

$$\sum_{e \in E^+(k)} x_e^{ij} - \sum_{e \in E^-(k)} x_e^{ij} = \begin{cases} \alpha_j R_i + \alpha_i C_j & \text{if } k = i \\ -\alpha_j R_i - \alpha_i C_j & \text{if } k = j \\ 0 & \text{otherwise} \end{cases} \quad \forall i, j, k \in N \tag{1}$$

$$\sum_{i,j \in N} x_e^{ij} \leq u_e \forall e \in E \tag{2}$$

$$\alpha_i \geq 0 \forall i \in N \tag{3}$$

$$x_e^{ij} \geq 0 \forall e \in E, \forall i, j \in N. \tag{4}$$

Constraints (1) corresponding to the routing of $\alpha_j R_i + \alpha_i C_j$ amount of flow from node i to node j. Constraints (2) are the link capacity constraints. By using per-source flow variables x_e^i instead of per source–destination variables x_e^{ij}, the number of flow variables in the above linear program can be reduced by a factor of n.

The above linear program is of polynomial size and is amenable for solution with LP solvers like CPLEX [5]. However, it is well known that running times of general linear programming based algorithms for network problems do not scale well with increasing network size.%beyond few tens of nodes. So we propose to design a fast combinatorial algorithm (FPTAS) with performance guarantees for the problem.

B. Incorporating Node Capacity Constraints in IP-Over-Optical Networks

Consider the deployment of our routing scheme in IP-over-optical networks as discussed in Section VI-A. The end-to-end IP traffic traverses router-to-OXC links not only at the source and destination nodes but also at the intermediate nodes. This

router-to-OXC traffic at a node is bounded by the aggregate connectivity of the IP router to the OXC at that node. Thus, we need to model such node capacity constraints in our problem formulation.

This is done by transforming the graph representation of the network as follows. Split each node into two sub-nodes, one representing the IP router and another the OXC at that node. All links incident at each node in the original graph are now incident at corresponding OXC sub-node. Add links in either direction connecting the router and OXC sub-nodes with capacity equal to the given router-to-OXC connectivity at that node. Traffic originates and terminates at the router sub-nodes in this transformed graph. Transit traffic traverses the OXC sub-nodes only, except at the intermediates nodes where it uses the router-to-OXC links to enter and leave the router sub-nodes. With this graph transformation, we can apply the problem formulation from Section VII-A in the context of IP-over-optical networks.

C. Path Flow Based Formulation

In this section, we present a path indexed linear programming formulation for the above problem. This will be subsequently used to develop the fast combinatorial algorithm (FPTAS) in Section VIII.

Let $\mathcal{P}_{ij}$ denote the set of all paths from node i to node j. Let $x(P)$ denote the traffic on path P.

$$\text{maximize} \sum_{i \in N} \alpha_i$$

subject to

$$\sum_{P \in \mathcal{P}_{ij}} x(P) = \alpha_j R_i + \alpha_i C_j \quad \forall i, j \in N \quad (5)$$

$$\sum_{P \ni e} x(P) \leq u_e \quad \forall e \in E \quad (6)$$

$$\alpha_i \geq 0 \quad \forall i \in N \quad (7)$$

$$x(P) \geq 0 \quad \forall P. \quad (8)$$

In Section VIII, we state the dual of the linear program. In general, a network can have an exponential number of paths (in the size of the network). Hence, this (primal) linear program can have possibly exponential number of variables and its dual can have an exponential number of constraints—they are both not suitable for running on medium to large sized networks. The usefulness of the primal and dual formulation is in designing a fast combinatorial algorithm for the problem.

VIII. Fast Combinatorial Algorithm

In this section, we develop a fast combinatorial algorithm (FPTAS) that computes the traffic split ratios and routing of Phase 1 and Phase 2 paths up to $(1+\epsilon)$-factor of the optimal objective function value (maximum throughput) for any $\epsilon > 0$. We begin with the dual formulation of the linear program discussed above. The primal-dual approach we develop is adapted from the technique in Garg and Könemann [8] for solving the maximum multicommodity flow problem, where flows are augmented in the primal solution and dual variables are updated in an iterative manner.

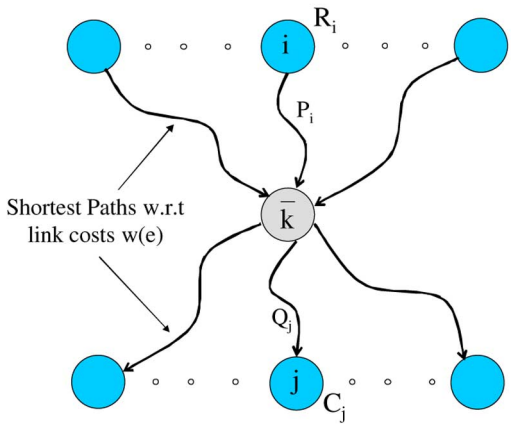

Fig. 4. One step in the primal-dual computation.

The dual formulation of the linear program outlined in Section VII-C associates a variable π_{ij} with each demand constraint in (3), and a non-negative variable $w(e)$ with each link capacity constraint in (4). Let $SP(i,j)$ denote the cost of the shortest path $P \in \mathcal{P}_{ij}$ under weights $w(e)$. That is,

$$SP(i,j) = \min_{P \in \mathcal{P}_{ij}} \sum_{e \in P} w(e).$$

After simplification and removal of the dual variables π_{ij}, the dual linear program can be written as below:

$$\text{minimize} \sum_{e \in E} u_e w(e)$$

subject to

$$\sum_{i \neq k} R_i SP(i,k) + \sum_{j \neq k} C_j SP(k,j) \geq 1 \quad \forall k \in N \quad (9)$$

$$w(e) \geq 0 \quad \forall e \in E. \quad (10)$$

For a given node k and weights $w(e)$, let $V(k)$ denote the left-hand side (LHS) of constraint (5). Given the weights $w(e)$, note that the values $V(k)$ for all $k \in N$ can be computed in polynomial time using a single all-pairs shortest path computation with link costs $w(e)$.

The algorithm works as follows. Start with initial weights $w(e) = \delta/u_e$ for all $e \in E$ (the quantity δ depends on ϵ and is derived later). Repeat the following until the dual objective function value becomes greater than 1:

1) Compute the node k for which $V(k)$ is minimum. This identifies a node $\bar{k}$ as well as paths P_i from node i to node $\bar{k}$ for all $i \neq \bar{k}$ and paths Q_j from node $\bar{k}$ to node j for all $j \neq \bar{k}$. (These are the corresponding shortest paths used in evaluating $V(\bar{k})$ as described above.) This is illustrated in Fig. 4.
2) For a traffic split ratio of 1 for intermediate node $\bar{k}$, the traffic on path P_i is R_i for all $i \neq \bar{k}$ and the traffic on path Q_j is C_j for all $j \neq \bar{k}$. Using this, compute the traffic $f(e)$ on link e per unit split ratio $\alpha_{\bar{k}}$ for intermediate node $\bar{k}$ as

$$f(e) = \sum_{i \neq \bar{k}, P_i \ni e} R_i + \sum_{j \neq \bar{k}, Q_j \ni e} C_j \quad \forall e \in E. \quad (11)$$

3) Compute the maximum value α for the traffic split ratio for intermediate node $\bar{k}$ that is consistent with (original) link capacity constraints for sending flow along paths P_i, Q_j as

$$\alpha = \min_{e \in E} \frac{u_e}{f(e)}. \tag{12}$$

4) For this value α of the split ratio for intermediate node $\bar{k}$, send αR_i amount of flow from node i to node $\bar{k}$ along path P_i for all $i \neq \bar{k}$ and αC_j amount of flow from node $\bar{k}$ to node j along path Q_j for all $j \neq \bar{k}$. Compute the total flow on link e is $\Delta(e) = \alpha f(e)$ for all $e \in E$.
5) Update the weights $w(e)$ as

$$w(e) \leftarrow w(e)\left(1 + \frac{\epsilon \Delta(e)}{u_e}\right) \quad \forall e \in E.$$

6) Increment the split ratio $\alpha_{\bar{k}}$ associated with node $\bar{k}$ by α.

When the above procedure terminates, primal capacity constraints on each link will be violated, since we were working with the original (and not residual) link capacities at each stage. To remedy this, we scale down the flows and traffic split ratios α_i uniformly so that capacity constraints are obeyed.

Note that since the algorithm maintains primal and dual solutions at each step, the optimality gap can be estimated by computing the ratio of the primal and dual objective function values. The computation can be terminated immediately after the desired closeness to optimality is achieved.

The pseudo-code for the above procedure, called Algorithm MAX-THROUGHPUT, is provided in the box on this page. Array $flow(e)$ keeps track of the traffic on link e as the algorithm progresses. The variable G is initialized to 0 and remains <1 as long as the dual constraints remain unsatisfied. After the **while** loop terminates, the maximum factor by which the capacity constraint is violated on any link is computed into $scale$. Finally, the α_i values are divided by the maximum capacity violation factor and the resulting values output.

The values of ϵ and δ are related, in the following theorem, to the approximation factor guarantee of Algorithm MAX-THROUGHPUT.

Theorem 1: For any given $\epsilon' > 0$, Algorithm MAX-THROUGHPUT computes a solution with objective function value within $(1+\epsilon')$-factor of the optimum for

$$\delta = \frac{1+\epsilon}{[(1+\epsilon)m]^{1/\epsilon}} \text{ and } \epsilon = 1 - \frac{1}{\sqrt{1+\epsilon'}}.$$

We end this section with a bound on the running time of Algorithm MAX-THROUGHPUT.

Theorem 2: For any given $\epsilon > 0$ chosen to provide the desired approximation factor guarantee in accordance with Theorem 1, Algorithm MAX-THROUGHPUT runs in time polynomial in the input size and $1/\epsilon$, that is,

$$O\left(\frac{nm}{\epsilon^2}(m + n\log n)\log m\right).$$

Proofs of the above theorems are provided in Appendix A.

IX. Evaluation on ISP Topologies

In this section, we evaluate the performance of two-phase routing. We first define a quantity called *throughput efficiency* that will be used to measure the effectiveness of two-phase routing against a general class of routing schemes that can handle traffic variability.

Algorithm MAX-THROUGHPUT:

$\alpha_k \leftarrow 0 \quad \forall\, k \in N$;
$w(e) \leftarrow \delta/u_e \quad \forall\, e \in E$;
$flow(e) \leftarrow 0 \quad \forall\, e \in E$;
$D \leftarrow 0$;

while $D < 1$ **do**
  Compute shortest path of cost $SP(i,j)$ from i to j under link costs $w(e) \ \forall\, i,j \in N$;
  $V(k) \leftarrow \sum_{i \neq k} R_i SP(i,k) + \sum_{j \neq k} C_j SP(k,j)$;
  $\bar{k} \leftarrow \arg\min_{k \in N} V(k)$;
  (Denote shortest path from i to $\bar{k}$ by P_i for all i and shortest path from $\bar{k}$ to j by Q_j for all j.)
  $f(e) \leftarrow \sum_{i \neq \bar{k}, P_i \ni e} R_i + \sum_{j \neq \bar{k}, Q_j \ni e} C_j \quad \forall\, e \in E$
  $\alpha \leftarrow= \min_{e \in E} \frac{u_e}{f(e)}$;
  Send αR_i flow on path P_i for all i and αC_j flow on path Q_j for all j and compute resulting capacity usage $\Delta(e)$ on link e for all e ;
  $flow(e) \leftarrow flow(e) + \Delta(e)$ for all e ;
  $w(e) \leftarrow w(e)(1 + \epsilon\Delta(e)/u_e)$ for all e ;
  $\alpha_{\bar{k}} \leftarrow \alpha_{\bar{k}} + \alpha$;
  $D \leftarrow \sum_{e \in E} u_e w(e)$;
end while

$scale \leftarrow \max_{e \in E} flow(e)/u_e$ for all $e \in E$;
$\alpha_k \leftarrow \alpha_k / scale$ for all $k \in N$;
Output traffic split ratios α_k for all $k \in N$;

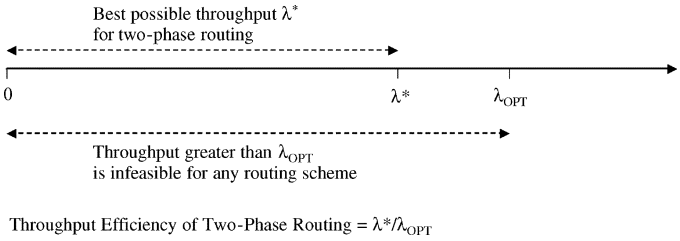

Fig. 5. Schematic illustrating throughput efficiency of two-phase routing.

A. Throughput Efficiency

Given a network with link capacities u_e and bounds R_i, C_j on the traffic matrix, an output λ^* of the problem formulation in Section VII provides a guarantee that all matrices in $\lambda^* \cdot \mathcal{T}(\vec{R}, \vec{C})$ can be routed by two-phase routing. The highest possible throughput λ_{OPT} is admitted by the optimal scheme among the class of schemes that is allowed to make the routing dynamically dependent on the traffic matrix. This is illustrated in Fig. 5. The ratio $\lambda^*/\lambda_{\mathrm{OPT}}$ (≤ 1) is defined as the *throughput efficiency* of two-phase routing.

Note that the throughput efficiency, as defined above, is different from the oblivious ratio of Azar *et al.* [2]. In the latter case, the routing is compared with the *best routing for a single traffic matrix*. In our case, the routing is compared with *best scheme for routing all matrices in* $\mathcal{T}(\vec{R}, \vec{C})$.

TABLE I
ROCKETFUEL TOPOLOGIES: ORIGINAL NUMBER OF ROUTERS AND INTER-ROUTER LINKS, AND NUMBER OF COALESCED POPS AND INTER-POP LINKS

Topology	Routers (original)	Links (inter-router)	PoPs (coalesced)	Links (inter-PoP)
Telstra (Australia) 1221	108	306	57	59
Sprintlink (US) 1239	315	1944	44	83
Ebone (Europe) 1755	87	322	23	38
Tiscali (Europe) 3257	161	656	50	88
Exodus (Europe) 3967	79	294	22	37
Abovenet (US) 6461	141	748	22	42

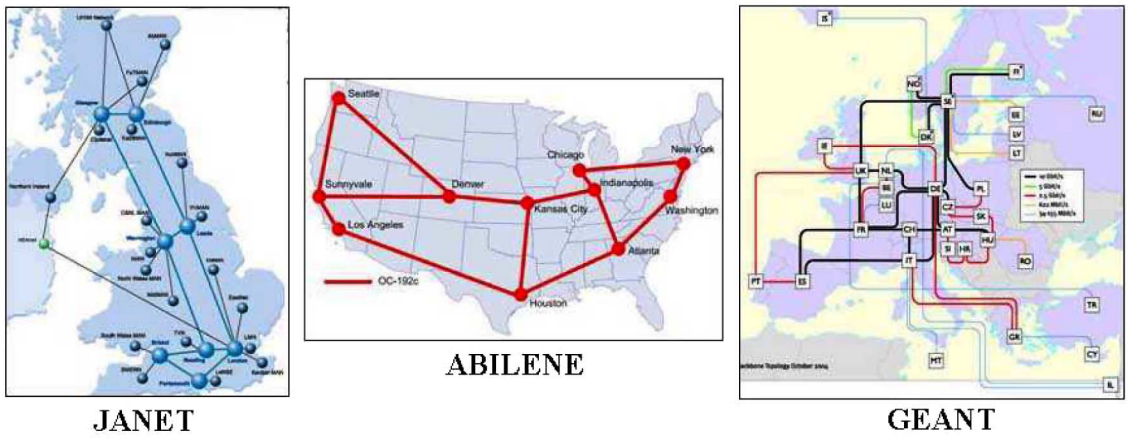

Fig. 6. Three research network topologies.

It is shown in [12] that the throughput efficiency of two-phase routing is at least 0.5 (or, 50%) when the ingress–egress capacities are symmetric, i.e., $R_i = C_i$ for all i. The latter assumption holds for all the ISP topologies we use in our experiments because network routers and switches have bidirectional ports (line cards). We will see that the throughput efficiency of two-phase routing on the evaluated topologies is significantly better than this theoretical lower bound of 50%.

The value λ_{OPT} is *co*$\mathcal{NP}$-hard to compute [18]. Suppose that we take any single matrix $T \in \mathcal{T}(\vec{R}, \vec{C})$ and compute the maximum multiplier $\lambda(T)$ (using a maximum concurrent flow formulation [20]) such that $\lambda(T) \cdot T$ can be feasibly routed in the network with given link capacities. Then, $\lambda_{\mathrm{OPT}} \leq \lambda(T)$, and hence $(\lambda^*/\lambda(T)) \leq (\lambda^*/\lambda_{\mathrm{OPT}}) \leq 1$. Thus, for any traffic matrix $T \in \mathcal{T}(\vec{R}, \vec{C})$, the quantity $\lambda^*/\lambda(T)$ is a lower bound on the throughput efficiency of two-phase routing. To obtain a tight lower bound, we would a like to identify a matrix $T \in \mathcal{T}(\vec{R}, \vec{C})$ for which $\lambda(T)$ is minimum. This matrix T is hard to compute. In Appendix B, we describe a heuristic approach to find a matrix that gives tight lower bounds.

B. Topologies and Link/Ingress-Egress Capacities

For our experiments, we use six ISP topologies collected by Rocketfuel, an ISP topology mapping engine [21]. These topologies list multiple intra-PoP (Point of Presence) routers and/or multiple intra-city PoPs as individual nodes. We coalesced PoPs into nodes corresponding to cities so that the topologies represent geographical PoP-to-PoP ISP topologies. Some data about the original Rocketfuel topologies and their coalesced versions is provided in Table I.

We also use three research network topologies, namely, the UK research network JANET,[1] the US research backbone ABILENE,[2] and the European research network GEANT.[3] These are shown in Fig. 6.

Link capacities, which are required to compute the maximum throughput, are not available for the Rocketfuel topologies. Rocketfuel computed OSPF/IS-IS link weights for the topologies so that shortest cost paths match observed routes. In order to deduce the link capacities from the weights, we assumed that the given link weights are the default setting for OSPF weights in Cisco routers, i.e., inversely proportional to the link capacities [4]. The link capacities obtained in this manner turned out to be symmetric, i.e., $u_{ij} = u_{ji}$ for all $(i, j) \in E$.

There is also no available information on the ingress–egress traffic capacities at each node for the Rocketfuel topologies. Because ISPs commonly engineer their PoPs to keep the ratio of add/drop and transit traffic approximately fixed, we assumed that the ingress–egress capacity at a node is proportional to the total capacity of network links incident at that node. We also assume that $R_i = C_i$ for all nodes i—since network routers and switches have bidirectional ports (line cards), hence the ingress and egress capacities are equal. Thus, we have $R_i\ (= C_i) \propto \sum_{e \in E^+(i)} u_e$.

In contrast, link capacities are available for all three research network topologies. Ingress-egress capacities are available for the JANET topology only.

[1]http://www.ja.net

[2]http://abilene.internet2.edu

[3]http://www.geant.net

TABLE II
THROUGHPUT EFFICIENCY OF TWO-PHASE ROUTING AND POINT-TO-POINT PIPE MODEL

Topology	Throughput Efficiency of Two-Phase Routing	Throughput Efficiency of Point-to-Point Pipe Model
Telstra (Australia) 1221	100%	5.39%
Sprintlink (US) 1239	97.71%	3.76%
Ebone (Europe) 1755	98.90%	7.33%
Tiscali (Europe) 3257	95.65%	5.97%
Exodus (Europe) 3967	100%	13.15%
Abovenet (US) 6461	94.82%	10.44%
JANET (8-node)	96.56%	36.69%
ABILENE (11-node)	95.47%	16.85%
GEANT (33-node)	97.35%	11.03%

C. Experiments and Results

To obtain the maximum throughput for two-phase routing for purposes of comparison with that of the optimal scheme, we used the exact linear programming formulation from Section VII and solved it using CPLEX [5].

1) Throughput Efficiency: In Table II, we list the throughput efficiency of two-phase routing for the six Rocketfuel and three research network topologies. We compare this with the throughput efficiency of the *point-to-point pipe provisioning model* in which a fixed demand of $\min(R_i, C_j)$ is provisioned from node i to node j for all $i, j \in N$ to handle the maximum possible traffic from i and j under the given ingress–egress capacities. Similar to that for two-phase routing, the throughput efficiency of the point-to-point pipe model is measured relative to the throughput of the optimal scheme. The throughput of the point-to-point pipe provisioning scheme is computed using a multi-commodity flow based linear programming formulation.

Table II clearly shows that the throughput of two-phase routing is very close to that of the best possible scheme for routing with traffic variability on all nine topologies. Thus, two-phase routing, surprisingly, is able to meet the requirements of Section III without any appreciable decrease in throughput compared to the optimal scheme. Table II also brings out the poor throughput performance of the point-to-point pipe model, the throughput efficiency of which is much smaller.

2) Number of Intermediate Nodes: In Table III, we list the number of intermediate nodes i with $\alpha_i > 0$ for maximum throughput two-phase routing on the six Rocketfuel and three research network topologies. Interestingly, the number of such intermediate nodes, especially for the larger topologies, is small compared to the total number of nodes. This may have favorable implications in the adaptation of the scheme to specialized service overlays and middlebox routing as explained in Section VI. In these two application scenarios, the intermediate nodes are sites for locating overlay routing servers and middleboxes, respectively.

3) Equal versus Unequal Traffic Split Ratios: For the two-phase routing scheme, we denote the throughput for equal traffic split ratios by λ_{equal} and the throughput for our general problem formulation that allows unequal traffic split ratios by λ_{unequal}. It is easy to see that $\lambda_{\text{unequal}} \geq \lambda_{\text{equal}}$. In Table IV, we give the throughput of two-phase routing with equal and unequal split ratios. The percentage increase in throughput $\lambda_{\text{unequal}} - \lambda_{\text{equal}}/\lambda_{\text{equal}}$ when we go from equal to unequal split ratios is also shown. When either the link capacities or ingress–egress capacities are scaled by a constant, the throughput values are scaled by the same constant. Hence, for comparison purposes, we have normalized the values so that the throughput for the unequal traffic split ratios case is $\lambda_{\text{unequal}} = 1.0$.

TABLE III
NUMBER OF INTERMEDIATE NODES IN TWO-PHASE ROUTING

Topology	Number of Intermediate Nodes
Telstra (Australia) 1221	1
Sprintlink (US) 1239	5
Ebone (Europe) 1755	4
Tiscali (Europe) 3257	7
Exodus (Europe) 3967	3
Abovenet (US) 6461	7
JANET (8-node)	3
ABILENE (11-node)	2
GEANT (33-node)	3

TABLE IV
THROUGHPUT OF TWO-PHASE ROUTING WITH UNEQUAL AND EQUAL TRAFFIC SPLIT RATIOS

Topology	$\lambda_{unequal}$	λ_{equal}	$\frac{\lambda_{unequal}-\lambda_{equal}}{\lambda_{equal}}$
Telstra (Australia) 1221	1.0	0.7756	28.93%
Sprintlink (US) 1239	1.0	0.3978	151.38%
Ebone (Europe) 1755	1.0	0.6137	62.95%
Tiscali (Europe) 3257	1.0	0.6625	50.95%
Exodus (Europe) 3967	1.0	0.8908	12.26%
Abovenet (US) 6461	1.0	0.7098	40.89%
JANET (8-node)	1.0	0.9910	9.08%
ABILENE (11-node)	1.0	0.8684	15.15%
GEANT (33-node)	1.0	0.0594	1583%

The results clearly bring out the increase in network throughput when the split ratios α_i are allowed to be unequal. The average savings for the nine topologies is 217% and the range is from 12% to as high as 1583%. We conclude that by allowing the traffic split ratios to be unequal, network throughput for two-phase routing can be increased significantly over the equal traffic split ratios case.

X. RELATED WORK

In Section IV, we reviewed related work in [2], [14] for direct source–destination routing along fixed paths. We pointed out two aspects of these approaches that do not meet the requirements of application scenarios discussed in Section III, namely (i) the source needs to *know the final destination of a packet* for routing it, and (ii) the bandwidth requirements of the (fixed) paths change with traffic variations.

Because of (i), these methods cannot be used in specialized service overlay models like i3 where the *final destination of a packet is not known at the source*. Because of (ii), the adaptation of these methods for IP-over-optical networks necessitates detection of changes in traffic patterns and dynamic reconfiguration of the provisioned optical layer circuits in response to it, a functionality that is not present in current IP-over-optical network deployments.

The origins of two-phase routing can be traced back to Valiant's randomized scheme for communication among parallel processors interconnected in a hypercube topology [24],

where routing is through a *randomly and uniformly chosen intermediate node* and the *split ratios are implicitly equal*. The two-phase routing scheme can be viewed as a deterministic scheme with possibly *unequal traffic split ratios* which can accommodate all traffic matrices within the network's natural ingress–egress capacity constraints. Our work considers many new aspects arising from its potential application to routing Internet traffic in ISP backbone networks.

Our current work is a sequel to [9]. In [25], a version of the scheme with *equal traffic split ratios of* $1/n$ and *equal ingress–egress capacities* ($R_i = C_i = c$ for all i) is considered, and subsequently extended to unequal traffic split ratios in [26]. The authors in [25] assume that the IP layer topology is a full-mesh (fully connected complete graph), so that the Phase 1 and Phase 2 paths are one hop in length. These paths need to be routed (via multi-hop paths) on the physical WDM topology (which is a sparse graph). Also, if the IP topology is not full-mesh, the Phase 1 and Phase 2 paths will be multi-hop at the IP layer itself. Our problem formulation for two-phase routing in [9] (and in this paper) models the multi-hop routing of Phase 1 and Phase 2 paths and can be applied to a general IP layer topology and a physical WDM topology. The network design version of the problem on uncapacitated topologies is considered in [18] and [26].

XI. Conclusion

The two-phase routing scheme was recently proposed for routing highly dynamic and changing traffic patterns on the Internet with QoS guarantees. If deployed, it will allow service providers to operate their networks in a quasi-static manner where both intra-domain paths and the bandwidths allocated to these paths is robust to extreme traffic variation. The scheme has the desirable properties of supporting (i) indirection in specialized service overlay models like i3, and (ii) static optical layer provisioning in IP-over-optical networks. To our knowledge, this routing scheme is the only one that is sufficiently versatile to handle the needs of such diverse applications that we studied while also being robust to extreme traffic fluctuation.

In this paper, we developed linear programming formulations and a fast combinatorial algorithm for routing under the scheme so as to maximize throughput. We compared the throughput performance of two-phase routing with that of the optimal scheme among the class of all schemes that are allowed to make the routing dynamically dependent on the traffic matrix. Experiments on actual ISP topologies taken from the Rocketfuel project show that the throughput of two-phase routing with intermediate node dependent traffic split ratios is within 6% of the optimal scheme on all evaluated topologies. Thus, two-phase routing achieves its robustness to traffic variation and its versatility in being applicable to the discussed networking scenarios without any significant over-provisioning of capacity.

Appendix

A. Proofs of Theorems 1 and 2

In this section, we provide proofs for Theorems 1 and 2 from Section VIII for the approximation factor guarantee and running time of Algorithm MAX-THROUGHPUT. We begin with some notation, then state some useful lemmas, and finally conclude with the proofs of the main theorems.

Given a set of dual weights $w(e)$, let $D(w)$ denote the dual objective function value and let $\Gamma(w)$ denote the minimum value of the LHS of dual program constraint (5) over all nodes $k \in N$. Then, solving the dual program is equivalent to finding a set of weights $w(e)$ such that $D(w)/\Gamma(w)$ is minimized. Denote the optimal objective function value of the latter by θ, i.e., $\theta = \min_w D(w)/\Gamma(w)$.

We introduce some more notation before stating an important lemma. Let w_{t-1} denote the weight function at the beginning of iteration t of the **while** loop, and let A_{t-1} be the value of $\sum_{j \in N} \alpha_j$ (primal objective function) up to the end of iteration $t-1$. Suppose the algorithm terminates after iteration L.

Lemma 1: At the end of every iteration t, $1 \le t \le L$ of Algorithm MAX-THROUGHPUT,the following holds:

$$D(w_t) \le m\delta \prod_{j=1}^{t} \left[1 + \frac{\epsilon}{\theta}(A_j - A_{j-1})\right].$$

Proof: Let $k = \bar{k}$ be the node for which $V(k)$ is minimum and P_i, Q_j be the corresponding paths (as defined earlier) along which flow is augmented during iteration t. Recall that the weights are updated as

$$w_t(e) = w_{t-1}(e)\left(1 + \frac{\epsilon\Delta(e)}{u_e}\right) \quad \forall e \in E$$

where $\Delta(e)$ is the total flow sent on link e during iteration t. Using this, we have

$$\begin{aligned} D(w_t) &= \sum_{e \in E} u_e w_t(e) \\ &= \sum_{e \in E} u_e w_{t-1}(e) + \epsilon \sum_{e \in E} w_{t-1}(e)\Delta(e) \\ &= D(w_{t-1}) \\ &\quad + \epsilon \sum_{e \in E} w_{t-1}(e)\left[\sum_{i \ne \bar{k}, P_i \ni e} \alpha R_i + \sum_{j \ne \bar{k}, Q_j \ni e} \alpha C_j\right] \\ &= D(w_{t-1}) \\ &\quad + \epsilon\alpha \sum_{e \in E} w_{t-1}(e)\left[\sum_{i \ne \bar{k}, P_i \ni e} R_i + \sum_{j \ne \bar{k}, Q_j \ni e} C_j\right]. \end{aligned}$$

Interchanging the summations on the right-hand side (RHS) of the above equation and first summing along links on paths P_i, Q_j, and then over i, j, respectively, we can rewrite the RHS of the above equation to obtain

$$\begin{aligned} D(w_t) &= D(w_{t-1}) \\ &\quad + \epsilon\alpha\left[\sum_{i \ne \bar{k}} R_i \sum_{e \in P_i} w_{t-1}(e) + \sum_{j \ne \bar{k}} C_j \sum_{e \in Q_j} w_{t-1}(e)\right] \\ &= D(w_{t-1}) + \epsilon\alpha\Gamma(w_{t-1}) \\ &= D(w_{t-1}) + \epsilon(A_t - A_{t-1})\Gamma(w_{t-1}). \end{aligned}$$

Using this for each iteration down to the first one, we have

$$D(w_t) = D(w_0) + \epsilon \sum_{j=1}^{t} (A_j - A_{j-1})\Gamma(w_{j-1}). \qquad (13)$$

From the definition of θ, we have $\theta \leq (D(w_{j-1})/\Gamma(w_{j-1}))$, whence $\Gamma(w_{j-1}) \leq (1/\theta)D(w_{j-1})$. Also, $D(w_0) = m\delta$. Using these in (13), we have

$$D(w_t) \leq m\delta + \frac{\epsilon}{\theta}\sum_{j=1}^{t}(A_j - A_{j-1})D(w_{j-1}). \qquad (14)$$

The property claimed in the lemma can now be proved using inequality (14) and mathematical induction on the iteration number t. We omit the details here, but point out that the induction basis case (iteration $t = 1$) holds since $w_0(e) = (\delta/u_e)\ \forall e \in E$ and $D(w_0) \leq m\delta$. ∎

We now estimate the factor by which the objective function value value A_L in the primal solution when the algorithm terminates needs to be scaled to ensure that link capacity constraints are not violated.

Lemma 2: When Algorithm MAX-THROUGHPUT terminates, the primal solution needs to be scaled by a factor of at most $\log_{1+\epsilon}((1+\epsilon)/\delta)$ to ensure primal feasibility.

Proof: Consider any link e and associated weight $w(e)$. The value of $w(e)$ is updated when flow is augmented on edge e. Let the sequence of flow augmentations (per iteration) on link e be $\Delta_1, \Delta_2, \ldots, \Delta_r$, where $r \leq L$. Let $\sum_{t=1}^{r}\Delta_t = \kappa u_e$, i.e., the total flow routed on link e exceeds its capacity by a factor of κ.

Because of the way in which α is chosen in accordance with (12), we have $\Delta_i \leq u_e$ for all i. Hence, the dual weight $w(e)$ is updated by a factor of at most $1+\epsilon$ after each iteration. Since the algorithm terminates when $D(w) \geq 1$, and since dual weights are updated by a factor of at most $1+\epsilon$ after each iteration, we have $D(w_L) < 1+\epsilon$. Since the weight $w(e)$, with coefficient u_e, is one of the summing components of $D(w)$, we have $u_e w_L(e) < 1+\epsilon$. Also, the value of $w_L(e)$ is given by

$$w_L(e) = \delta\prod_{t=1}^{r}\left(1 + \frac{\Delta_t}{u_e}\epsilon\right).$$

Using the inequality $(1+cx) \geq (1+x)^c\ \forall x \geq 0$ and any $0 \leq c \leq 1$ and setting $x = \epsilon$ and $c = (\Delta_t/u_e) \leq 1$, we have

$$\begin{aligned}\frac{1+\epsilon}{u_e} > w_L(e) &\geq \delta\prod_{t=1}^{r}(1+\epsilon)^{\Delta_t/u_e}\\ &= \delta(1+\epsilon)^{\sum_{t=1}^{r}\Delta_t/u_e}\\ &= \delta(1+\epsilon)^{\kappa}\end{aligned}$$

whence

$$\kappa < \log_{1+\epsilon}\frac{1+\epsilon}{\delta}.$$

∎

Proof of Theorem 1: Using Lemma 1 and the inequality $1 + x \leq e^x\ \forall x > 0$, we have

$$\begin{aligned}D(w_t) &\leq m\delta\prod_{j=1}^{t}e^{\frac{\epsilon}{\theta}(A_j - A_{j-1})}\\ &= m\delta e^{\epsilon A_t/\theta}.\end{aligned}$$

The simplification in the above step uses telescopic cancellation of the sum $(f_j - f_{j-1})$ over j. Since the algorithm terminates after iteration L, we must have $D(w_L) \geq 1$. Thus,

$$1 \leq D(w_L) \leq m\delta e^{\epsilon A_t/\theta}$$

whence

$$\frac{\theta}{A_L} \leq \frac{\epsilon}{\ln\frac{1}{m\delta}}. \qquad (15)$$

From Lemma 2, the objective function value of the feasible primal solution after scaling is at least

$$\frac{A_L}{\log_{1+\epsilon}\frac{1+\epsilon}{\delta}}.$$

The approximation factor for the primal solution is at most the gap (ratio) between the primal and dual solution. Using (15), this is given by

$$\begin{aligned}\frac{\theta}{A_L} &\leq \frac{\epsilon\log_{1+\epsilon}\frac{1+\epsilon}{\delta}}{\ln\frac{1}{m\delta}}\\ &= \frac{\epsilon}{\ln(1+\epsilon)}\frac{\ln\frac{1+\epsilon}{\delta}}{\ln\frac{1}{m\delta}}.\end{aligned}$$

The quantity $\ln(1+\epsilon/\delta)/\ln(1/m\delta)$ equals $1/(1-\epsilon)$ for $\delta = (1+\epsilon)/[(1+\epsilon)m]^{1/\epsilon}$. Using this value of δ, the approximation factor is upper bounded by

$$\frac{\epsilon}{\ln(1+\epsilon)}\frac{1}{(1-\epsilon)} \leq \frac{\epsilon}{(\epsilon-\epsilon^2/2)(1-\epsilon)} \leq \frac{1}{(1-\epsilon)^2}.$$

Setting $1+\epsilon' = 1/(1-\epsilon)^2$ and solving for ϵ, we get the value of ϵ stated in the theorem. ∎

Proof of Theorem 2: We first consider the running time of each iteration of the algorithm during which a node $\bar{k}$ and associated paths P_i, Q_j are chosen to augment flow. Selection of this node and the paths involves an all-pairs shortest path computation which can be implemented in $O(nm + n^2\log n)$ time using Dijkstra's shortest path algorithm with Fibonacci heaps [1]. All other operations within an iteration are absorbed (up to a constant factor) by the time taken for this all-pairs shortest path computation, leading to a total of $O(n(m + n\log n))$ time per iteration.

We next estimate the number of iterations before the algorithm terminates. Recall that in each iteration, flow is augmented along paths P_i, Q_j corresponding to the maximum value of intermediate node split ratio α such that the total flow $\Delta(e)$ sent on link e during that iteration is at most u_e. Thus, for at least one link e, $\Delta(e) = u_e$ and $w(e)$ increases by a factor of $1+\epsilon$. Accordingly, with each iteration, we can associate a weight $w(e)$ which increases by a factor of $1+\epsilon$.

Consider the weight $w(e)$ for fixed $e \in E$. Since $w_0(e) = \delta/u_e$ and $w_L(e) \leq (1+\epsilon/u_e)$ (as deduced in the proof of Lemma 2), the maximum number of times that this weight can be associated with any iteration is

$$\log_{1+\epsilon}\frac{1+\epsilon}{\delta} = \frac{1}{\epsilon}(1+\log_{1+\epsilon}m) = O\left(\frac{1}{\epsilon}\log_{1+\epsilon}m\right).$$

Since there are a total of m weights $w(e)$, hence the total number of iterations is upper bounded by $O((m/\epsilon)\log_{1+\epsilon} m)$. Multiplying this by the running time per iteration, we obtain the overall algorithm running time as $O((nm/\epsilon)(m + n\log n)\log_{1+\epsilon} m) = O((nm/\epsilon^2)(m + n\log n)\log m)$. ■

B. Upper Bounding Throughput of Optimal Scheme

To obtain a tight lower bound on the throughput efficiency of two-phase routing, we would like to identify a matrix $T \in \mathcal{T}(\vec{R}, \vec{C})$ for which $\lambda(T)$ is minimum. This matrix T is hard to compute. Intuitively, such a matrix will take large bandwidth to route. Hence, we can approximate it by computing a matrix that takes the highest bandwidth to route *along shortest paths*. (Note that maximum throughput routing does not necessarily route along shortest paths, hence this approach is a heuristic.) Let d_{ij} denote the hop count of a shortest path from node i to j for all $i, j \in N$. Let $B(T)$ denote the least bandwidth for routing the matrix $T \in \mathcal{T}(\vec{R}, \vec{C})$. The matrix $T \in \mathcal{T}(\vec{R}, \vec{C})$ that maximizes $B(T)$ is computable in polynomial time using the following linear programming formulation:

$$\text{maximize} \quad \sum_{i,j \in N} d_{ij} t_{ij}$$

subject to

$$\sum_{j \in N, j \neq i} t_{ij} \leq R_i \quad \forall i \in N \tag{16}$$

$$\sum_{i \in N, i \neq j} t_{ij} \leq C_j \quad \forall j \in N \tag{17}$$

$$t_{ij} \geq 0 \quad \forall i, j \in N. \tag{18}$$

The required bandwidth $B(T)$ is the objective function of the linear program and and the ingress–egress traffic capacities that define $\mathcal{T}(\vec{R}, \vec{C})$ form the constraints. Using the traffic matrix $T \in \mathcal{T}(\vec{R}, \vec{C})$ from the solution of this linear program, we compute the throughput $\lambda(T)$ and obtain an upper bound on the throughput of the optimal scheme.

REFERENCES

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, NJ: Prentice Hall, Feb. 1993.

[2] Y. Azar, E. Cohen, A. Fiat, H. Kaplan, and H. Räcke, "Optimal oblivious routing in polynomial time," in *Proc. 35th ACM Symp. Theory of Computing (STOC)*, San Diego, CA, 2003, pp. 383–388.

[3] W. Ben-Ameur and H. Kerivin, "New economical virtual private networks," *Commun. ACM*, vol. 46, no. 6, pp. 69–73, Jun. 2003.

[4] Configuring OSPF. Cisco Systems Product Documentation. [Online]. Available: http://www.cisco.com/univercd/home/home.htm

[5] ILOG CPLEX. [Online]. Available: http://www.ilog.com

[6] N. G. Duffield, P. Goyal, A. G. Greenberg, P. P. Mishra, K. K. Ramakrishnan, and J. E. van der Merwe, "A flexible model for resource management in virtual private network," in *Proc. ACM SIGCOMM 1999*, Cambridge, MA, Aug. 1999, pp. 95–108.

[7] J. A. Fingerhut, S. Suri, and J. S. Turner, "Designing least-cost nonblocking broadband networks," *J. Algorithms*, vol. 24, no. 2, pp. 287–309, 1997.

[8] N. Garg and J. Könemann, "Faster and simpler algorithms for multicommodity flow and other fractional packing problems," in *Proc. 39th Annu. Symp. Foundations of Computer Science (FOCS)*, Palo Alto, CA, Nov. 1998, pp. 300–309.

[9] M. Kodialam, T. V. Lakshman, and S. Sengupta, "Efficient and robust routing of highly variable traffic," in *Proc. 3rd Workshop on Hot Topics in Networks (HotNets-III)*, San Diego, CA, Nov. 2004, 6 pp.

[10] M. Kodialam, T. V. Lakshman, J. B. Orlin, and S. Sengupta, "A versatile scheme for routing highly variable traffic in service overlays and IP backbones," in *Proc. IEEE INFOCOM 2006*, Barcelona, Spain, Apr. 2006, 12 pp.

[11] M. Kodialam, T. V. Lakshman, J. B. Orlin, and S. Sengupta, "Preconfiguring IP-over-optical networks to handle router failures and unpredictable traffic," *IEEE J. Sel. Areas Commun.*, Special Issue on Traffic Engineering for Multi-Layer Networks, Jun. 2007.

[12] M. Kodialam, T. V. Lakshman, and S. Sengupta, "Maximum throughput routing of traffic in the hose model," in *Proc. IEEE INFOCOM 2006*, Barcelona, Spain, Apr. 2006, 11 pp.

[13] M. Kodialam, T. V. Lakshman, and S. Sengupta, "Throughput guaranteed restorable routing without traffic prediction," in *Proc. IEEE ICNP 2006*, Santa Barbara, CA, Nov. 2006, pp. 137–146.

[14] A. Kumar, R. Rastogi, A. Silberschatz, and B. Yener, "Algorithms for provisioning VPNs in the hose model," in *Proc. ACM SIGCOMM 2001*, San Diego, Aug. 2001, pp. 135–146.

[15] C. Labovitz, A. Ahuja, and F. Jahanian, "Experimental study of internet stability and backbone failures," in *Proc. 29th Int. Symp. Fault-Tolerant Computing (FTCS)*, Madison, WI, Jun. 1999, pp. 278–285.

[16] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot, "Traffic matrix estimation: Existing techniques and new directions," in *Proc. ACM SIGCOMM 2002*, Pittsburgh, PA, Aug. 2002, pp. 161–174.

[17] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture," RFC 3031, Jan. 2001.

[18] S. Sengupta, "Efficient and robust routing of highly variable traffic," Ph.D. dissertation, Massachusetts Inst. Technol. (MIT), Cambridge, MA, Dec. 2005.

[19] S. Sengupta, D. Saha, and V. P. Kumar, "Switched optical backbone for cost-effective scalable core IP networks," *IEEE Commun. Mag.*, vol. 41, no. 6, pp. 60–70, Jun. 2003.

[20] F. Shahrokhi and D. Matula, "The maximum concurrent flow problem," *J. ACM*, vol. 37, no. 2, pp. 318–334, 1990.

[21] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, "Measuring ISP topologies with Rocketfuel," *IEEE/ACM Trans. Networking*, vol. 12, no. 1, pp. 2–16, Feb. 2004.

[22] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana, "Internet indirection infrastructure," in *Proc. ACM SIGCOMM 2002*, Pittsburgh, PA, Aug. 2002, pp. 73–86.

[23] D. Thaler and C. Hopps, "Multipath issues in unicast and multicast next-hop selection," RFC 2919.

[24] L. G. Valiant, "A scheme for fast parallel communication," *SIAM J. Comput.*, vol. 11, no. 7, pp. 350–361, 1982.

[25] R. Zhang-Shen and N. McKeown, "Designing a predictable internet backbone network," in *Proc. 3rd Workshop on Hot Topics in Networks (HotNets-III)*, San Diego, CA, Nov. 2004, 6 pp.

[26] R. Zhang-Shen and N. McKeown, "Designing a predictable internet backbone with valiant load-balancing," in *Proc. Int. Workshop on Quality of Service (IWQoS) 2005*, Passau, Germany, Jun. 2005, pp. 178–192.

Murali Kodialam (M'99–A'00) received the Ph.D. degree in operations research from Massachusetts Institute of Technology, Cambridge, in 1991.

He has been with Bell Labs, Murray Hill, NJ, since October 1991. He currently is in the High Speed Networks Research Department, working on resource allocation and performance of communication systems including routing in MPLS systems, topology construction and routing in ad hoc wireless networks, and reliable routing in optical networks.

Dr. Kodialam is a member of INFORMS.

T. V. Lakshman (M'85–SM'98–F'05) received the Master's degree in physics from the Indian Institute of Science, Bangalore, India, and the Ph.D. degree in computer science from the University of Maryland, College Park.

He is currently the Director of the Communication Protocols and Networking Research Department at Bell Labs, Alcatel-Lucent, Murray Hill, NJ. His research interests and contributions span a spectrum of networking topics including switch architectures, network design, TCP performance, traffic management, video transmission over packet networks and network security.

Dr. Lakshman has received several Best Paper Awards from the ACM and the IEEE. He was an Editor of the IEEE/ACM TRANSACTIONS ON NETWORKING from 1996 to 2002. He is currently an Editor of IEEE TRANSACTIONS ON MOBILE COMPUTING. He is a Fellow of the IEEE and ACM.

James B. Orlin is the Edward Pennell Brooks Professor of Operations Research at the MIT Sloan School. He has served as a Fulbright Fellow to The Netherlands, as an NSF Presidential Young Investigator, and as a co-director of the MIT Operations Research Center. He specializes in network and combinatorial optimization. With two colleagues, he has written a graduate-level text, *Network Flows: Theory, Algorithms, and Applications*. This text was the winner of the 1993 Lanchester Prize for the best English language publication in Operations Research. He is also interested in applications of network optimization and combinatorial optimization to logistics and vehicle routing.

Sudipta Sengupta (M'01–SM'07) received the B.Tech. degree in computer science and engineering from the Indian Institute of Technology (IIT), Kanpur, India, and the M.S. and Ph.D. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology (MIT), Cambridge, MA.

He is currently with Microsoft Research, Redmond, WA. Previously, he spent five years at Bell Laboratories, Lucent Technologies. He has published numerous research papers in some of the top conferences, journals, and technical magazines. He has taught advanced courses on networking at academic/research and industry conferences. He has authored patents in the area of computer networking. His research interests include algorithms/optimization, system architecture, protocols, and software for data networking, wireless networking, and network security.

Dr. Sengupta was awarded the President of India Gold Medal at IIT-Kanpur for graduating at the top of his class across all disciplines. His work on oblivious routing of Internet traffic won the IEEE Communications Society Leonard G. Abraham Prize in the Field of Communication Systems for 2008.