Coarse optical circuit switching by default, rerouting over circuits for adaptation

Jerry Chou* and Bill Lin

University of California San Diego, La Jolla, California 92093, USA
**Corresponding author: jchou@cs.ucsd.edu*

Received July 3, 2008; revised October 16, 2008; accepted November 10, 2008; published December 17, 2008 (Doc. ID 98177)

As Internet traffic continues to grow unabated at an exponential rate, it is unclear whether the existing packet-routing network architecture based on electronic routers will continue to scale at the necessary pace. On the other hand, optical fiber and switching elements have demonstrated an abundance of capacity that appears to be unmatched by electronic routers. In particular, the simplicity of circuit switching makes it well suited for optical implementations. We present what we believe to be a new approach to optical networking based on a paradigm of coarse optical circuit switching by default and adaptive rerouting over circuits with spare capacity. We consider the provisioning of long-duration quasi-static optical circuits between edge routers at the boundary of the network to carry the traffic by default. When the provisioned circuit is inadequate, excess traffic demand is rerouted through circuits with spare capacity. In particular, by adaptively load balancing across circuits with spare capacity, excess traffic is routed to its final destination without the need to create circuits on the fly. Our evaluations on two separate real, large Internet service provider point-of-presence-level topologies, Abilene and GEANT, show that only a very small amount of excess traffic needs to be rerouted even during peak traffic hours when the circuit configurations are carefully chosen and that this excess traffic could always be accommodated using our adaptive rerouting approach. We also demonstrate that our adaptive load-balancing approach is robust to sudden unexpected traffic changes by demonstrating its ability to reroute traffic under a number of hot-spot scenarios.

OCIS codes: 060.6718, 060.6719, 060.1155, 060.4250.

1. Introduction

The Internet has become the main conduit for virtually all wide-area data communications as it continues its phenomenal growth in traffic volumes and reach, extending into telephony and television broadcast services that were once only transported in the domain of dedicated networks. For the past decade, Internet traffic has been doubling nearly every year, and there is no indication that this rate of growth will decelerate in the near future. Although the packet-switching approach used in the Internet backbone networks has thus far been able to keep up, it is unclear whether electronic routers that have been used at the core of backbone networks will continue to scale to match future traffic growth or optical link rates [1].

On the other hand, optical fiber and switching elements have demonstrated an abundance of capacity that appears to be unmatched by electronic routers. The rate of increase in optical transport capacity has been keeping pace with traffic growth (with 100 Gbits/s per wavelength in the next generation). Thus, one possible way of keeping pace with future traffic demands is to build an all-optical backbone network. However, packet switching requires the buffering and processing of packets, of which optical switches are not capable today, and it is unclear whether these functions can be practically realized in optics. On the other hand, circuit switching has a much simpler data transport, making it well suited to optics and its vast capacity potential.

To harness the huge capacity of optical circuit switching in an evolutionary way that is compatible with packet switching at the edge of the network, transparent to the user, a number of candidate optical network data transport architectures have been proposed [2–13]. From a bird's-eye view, these architectures all share a similar conceptual starting point in which the core of the network is an all-optical circuit-switched cloud, as depicted in Fig. 1. The optical circuit-switched cloud is comprised of

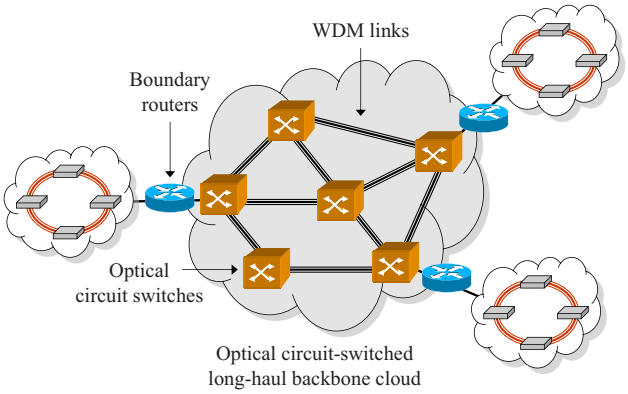

Fig. 1. Optical circuit-switched cloud with boundary routers.

long-haul dense wavelength division multiplexing (DWDM) links that are interconnected by optical cross connects (OXCs). Traffic traverses the circuit-switched cloud through pre-established circuits (lightpaths) at optical speeds. Boundary routers at the edge of the circuit-switched cloud provide a compatible packet-switching interface to the rest of the Internet. The different proposed optical network data transport architectures differ in how they adapt to changing traffic conditions and the corresponding requirements on the granularity of circuits and the frequency of changes to the circuit configurations.

Some approaches are based on frequent changes to circuit configurations [2–5,9,10]. For example, in optical burst switching (OBS) [2,3], bursts of data are aggregated at the network edge by the boundary routers, and an out-of-band signaling process is used for establishing temporary circuits across the optical circuit-switched cloud for each burst. OBS adapts to changing traffic conditions by changing the circuit configurations on a frequent time scale (i.e., at each data burst). In transmission control protocol (TCP) switching [4], the detection of a new application (TCP) flow triggers the creation of its own new circuit. TCP switching adapts to changing traffic conditions by frequent creations of new fine-grained circuits. A dynamic coarse-circuit-switching scheme has also been proposed (Chap. 5 of [5]) in which a coarse circuit is established between each pair of boundary routers for carrying traffic with the same pair of ingress–egress (IE) nodes. This dynamic coarse-circuit-switching approach adapts to changing traffic conditions by frequently adjusting the circuit configurations on relatively short time scales based on an online traffic estimation mechanism. Although the frequency of dynamic circuit reconfigurations imposed by the above approaches, of the order of tens of microseconds to a second, is well within the capabilities of available optical switching technologies, the coordination of such frequent networkwide reconfigurations is not easy. Moreover, new signaling mechanisms and (electronic) control planes are required to facilitate the coordination.

In this paper, we introduce COPLAR, a new optical networking design based on a new paradigm of coarse optical circuit switching by default and adaptive rerouting of excess traffic over circuits with spare capacity when necessary. COPLAR [14] stands for Coarse OPticaL circuit switching with Adaptive Rerouting. COPLAR exploits in part the observation that future traffic conditions can be predicted offline using past observations. Previous studies [15,16] have shown that the aggregate traffic at the core of the network tends to be very smooth and that it follows strong diurnal patterns that are easy to characterize. Figure 2 shows the total aggregate traffic on a backbone link in a tier-1 U.S. Internet service provider (ISP) backbone network [15]. As can be observed in the figure, the aggregate traffic varies over time in a regular and predict-

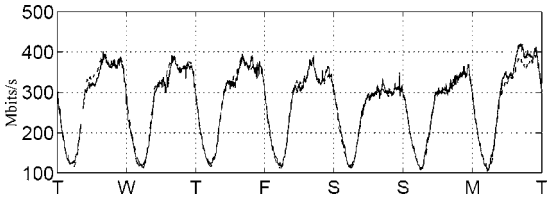

Fig. 2. Aggregate traffic on a tier-1 U.S. backbone link [15].

able way. Such diurnal traffic observations over repeated data sets suggest that coarse circuits could be provisioned to handle the expected traffic. Indeed, we make use of past traffic measurements to precompute offline such coarse circuit configurations. As we shall see in our extensive evaluations in Section 6, careful offline selection of coarse circuit configurations that take into account the statistical daily traffic variations over different hours observed in past measurements can produce coarse circuits that can accommodate the actual traffic most of the time. Therefore, in our approach, traffic is sent by default directly over the coarse optical circuit provisioned between the corresponding pair of boundary routers.

However, our approach also provides a mechanism for adapting to scenarios in which the actual traffic differs from prediction or when there are sudden unexpected changes in traffic (e.g., due to external events). In particular, our solution uses an adaptive load-balancing algorithm for rerouting (hopefully small amounts of) excess traffic over circuits that have spare capacity when the provisioned circuit provides insufficient capacity. By adaptively load balancing across circuits with spare capacity, excess traffic is routed to its final destination without the need to create new circuits on the fly—additional capacity needs are met through rerouting rather than fast dynamic circuit establishment. This is in notable contrast to dynamic circuit-provisioning approaches discussed above that rely on frequent circuit reconfigurations on relatively short time scales for adapting to changing traffic conditions. Intuitively, the approach works because the provisioned circuits form a logical topology over the boundary routers with many nondirect paths that can carry the excess traffic from a source to its final destination. Our adaptive load-balancing approach exploits the significant amount of path diversity available for traffic rerouting. As we shall see in Section 6, only a very small amount of excess traffic needs to be rerouted even during peak traffic hours when the circuit configurations are carefully chosen—under 7% in the worst case based on actual traffic for two real, large backbone networks. With adaptive rerouting, our approach was able to accommodate this excess traffic. We also demonstrate that our adaptive load-balancing approach is robust to sudden unexpected traffic changes by demonstrating its ability to reroute traffic under a number of hot-spot scenarios.

Although our proposed solution draws upon the pseudo-periodic behavior observed in real traffic to precompute offline circuit configurations, our approach differs from previous offline circuit configuration solutions in two important ways. First, previous offline circuit configuration approaches did not consider ways for adapting to unexpected traffic changes, which poses a legitimate concern for their deployment. For our adaptive load-balanced rerouting mechanism, we propose what we believe to be a novel approach for ensuring loop-free routing. One way to ensure loop-free routing is to limit consideration to paths with the minimum number of hops, but this policy does not match well with the logical topology formed by our circuits since the minimum number of hops will always be a direct circuit. Instead, we propose a novel efficient algorithm for constructing a maximal acyclic graph between each IE pair in the logical topology formed by the circuits, which provides a high degree of path diversity for rerouting.

Second, previous offline configuration approaches were designed to handle a specific traffic matrix or an entire set of traffic matrices [17–21]. Our work is different in that our formulation takes into consideration the statistical daily traffic variations observed in past measurements and the probability of traffic demands given their statistical distribution of occurrence in past measurements. In particular, we propose a new offline circuit configuration formulation that explicitly considers the statistical properties of past observations. In our formulation, the precomputed coarse circuit configurations do not necessarily provide sufficient circuit capacities for supporting all the traffic matrices captured in the historical data sets. Instead, our problem is formulated as a utility max–min fair bandwidth allocation problem that aims to maximize the acceptance probability of the expected traffic demand by using the cumulative distribution function over the historical data sets as the objective function. Our solution allocates all available network resources across multiple paths to provide as much headroom as possible. Since our solution does not rely on an online dynamic circuit creation mechanism, there is no need to leave behind network resources for establishing new circuits. Our adaptive rerouting mechanism can reroute traffic in a load-balanced manner in case of unexpected traffic changes or failures. To the best of our

knowledge, the general multipath utility max–min fair-bandwidth allocation problem has not been solved previously. Previous utility max–min fair allocation formulations only considered the single-path case [22–25], and previous multipath max–min fair allocation formulations did not consider general (nonlinear) utility functions [26]. An important contribution of this work is a first solution to this open general problem.

We extensively evaluated COPLAR on two real, large point-of-presence- (PoP-) level backbone networks, namely, Abilene [27] and GEANT [28], with real traffic trace data over 2 months. Our evaluations show a number of interesting results. First, the results show that our circuit-provisioning algorithm can produce circuit configurations that can accommodate the actual traffic demand most of the time, with at most 7% of the demand that needs to be rerouted in the worst case during peak traffic hours. With adaptive rerouting, all offered traffic demand could be accommodated. Second, when the actual traffic matrices are scaled up, COPLAR is surprisingly able to handle a higher traffic load than conventional packet routing [29,30]. The effectiveness of COPLAR can be attributed to the path diversity available for the adaptive rerouting of traffic. Finally, to evaluate the robustness of COPLAR against unexpected traffic changes, we evaluated COPLAR using a range of hot-spot traffic matrices. In comparison with conventional packet routing [29,30], COPLAR is able to achieve comparable throughput, even though the unexpected traffic patterns differ from those that were used to determine the circuit configurations. These results again demonstrate the robustness of adaptive rerouting for accommodating changing traffic conditions.

The rest of this paper is organized as follows. We first summarize additional related work in Section 2. We then introduce our circuit provisioning and rerouting methods in Section 3 and Section 4, respectively. Then we briefly describe our experimental setup in Section 5, and we present the results of our evaluation in Section 6. Finally, concluding remarks are presented in Section 7.

2. Additional Related Work

A number of related works have already been introduced in Section 1. As such, we restrict discussion in this section to related work not yet described to minimize redundancy. To implement the actual circuit configurations, a number of signaling protocols have been proposed. For example, generalized multiprotocol label switching (GMPLS) [11,12] has been proposed as a way to extend MPLS [31] to incorporate circuit switching in the domains of time, frequency, and space. GMPLS defines the signaling mechanisms for various management aspects of constructing coarse label-switched paths that are circuit switched. The scope of GMPLS is complementary to ours in that it does not specify a control algorithm to decide when to create circuits and with what capacity. Rather, GMPLS provides us with one way to implement our precomputed coarse circuit configurations.

Besides the dynamic circuit configuration approaches discussed in Section 1, another interesting transport architecture proposal is optical flow switching (OFS) [9,10]. The approach is different than the high-level design depicted in Fig. 1 in that the user-initiated data circuits are established between end users rather than boundary routers. Nonetheless, it provides another representative transport architecture for creating circuits on the fly, which can adapt to changing traffic conditions along with the necessary signaling mechanism and control plane to change circuit configurations on relatively short time scales.

Approaches based on fast optical packet switching have also been proposed (e.g., [32–34]). For example, in [34], the core of the optical network comprises high-performance optical packet routers with intelligent edge routers at the boundary of the optical network to perform intelligent traffic shaping.

Another approach based on long-duration coarse circuits is based on two-phase routing [6–8]. Rather than configuring circuits to support a specific set of traffic matrices, researchers have suggested the configuration of long-duration circuits that are optimized for the worst-case throughput under all possible traffic patterns permissible within the network's natural ingress–egress capacity constraints. These approaches work by setting up static coarse circuits and sending traffic across the optical circuit-switched cloud twice in a load-balanced manner in two phases. However, as we shall see in Section 6, optimizing for the worst case leads to rather pessimistic performance.

Finally, motivated by the desire to adapt to changing traffic conditions, possibly corresponding to sudden traffic shifts, a number of online adaptive routing policies have

been developed. TeXCP [35], MATE [36], and REPLEX [37] are representative examples. These approaches take advantage of alternative routing paths in the network. These adaptive routing approaches are complementary to our work in that we can incorporate them into our solution framework for routing over nondirect paths via traversal through multiple circuits. We currently employ a simple adaptive algorithm that simply reroutes based on the available spare circuit capacities seen from the immediate outgoing circuits at each node. The key problem that we have solved is the construction of a loop-free routing table that aims to maximize the available path diversity in the logical topology formed by the circuits, as discussed in Section 4. As we shall see in Section 6, we were able to robustly reroute excess traffic using our simple adaptive load-balancing algorithm even under a number of hot-spot traffic scenarios in which there is a substantial unexpected change in the traffic. More sophisticated adaptive routing algorithms such as TeXCP [35], MATE [36], and REPLEX [37] can be used with our proposed loop-free (nondirect) path construction method to achieve potentially better performance.

3. Coarse Circuit Configuration

3.A. Problem Formulation

Our circuit configuration problem is to find a bandwidth allocation vector $B(t)$ and its corresponding circuits $P(t)$ for each time interval t such that for any flow i we can set up a circuit with capacity $B_i(t)$ by reserving bandwidth along a set of paths $P_i(t)$ at time t. Flow i refers to the aggregate traffic between the corresponding IE router pair (s_i, d_i). Unless otherwise noted, we will use the terms flow and IE flow interchangeably. Also, for the remainder of the paper, unless otherwise noted, we will simply refer to a coarse circuit as a circuit. The formal definition of a circuit configuration is given in Definition 1, where the index t is omitted for simplicity:

Definition 1 [**Circuit Configuration** (B,P)]. *B is a bandwidth allocation vector whose element B_i is the rate assigned to the flow i. Each rate B_i can be routed through a set of paths P_i. For each $P \in P_i$, $f(P)$ represents the portion of B_i that has been split to P*:

$$\sum_{\forall P \in P_i} f(P) = B_i, \quad \forall \text{ flow } i. \tag{1}$$

Notice that in our problem definition, the paths are not given. Rather, path choices are variables in our problem. A feasible circuit configuration is one in which the circuit provisioning does not exceed any link capacity, as defined in Definition 2:

Definition 2 (Feasible Circuit Configuration). *Given a (physical) network topology (V,E) with link capacities C, where $C(e)$ is the capacity of link e, a circuit configuration (B,P) is said to be feasible if and only if the total bandwidth allocated to circuits passing each link e does not exceed the link's:*

$$\sum_{\forall P \ni e} f(P) \leq C(e), \quad \forall e \in E. \tag{2}$$

Although many possible feasible circuit configurations exist, we formulate the problem as a multipath utility max–min fair allocation problem in which we derive our circuit configurations to capture the traffic variance observed in historical traffic measurements by using a cumulative distribution function (CDF) model as the utility function. In particular, the use of CDFs [22] captures the acceptance probability of a particular bandwidth allocation as follows. Let $X_i(t)$ be a random variable that represents the actual traffic for flow i at time t, and let $x_i(t)$ be the bandwidth allocation. Then the CDF of $X_i(t)$ or the acceptance probability of flow i is denoted as

$$Pr[X_i(t) \leq x_i(t)] = \Phi_{i,t}[x_i(t)].$$

Therefore, the intuition of maximizing acceptance probability is to have a circuit provisioning that can carry the traffic of each IE pair at the maximum probability by considering the traffic variance and distribution of individual IE pairs.

In general, the expected traffic can be modeled using different probability density functions with the corresponding CDFs. One probability density function is to use the empirical distribution that directly corresponds to the historical traffic measurements taken. Let $[r_{i,t}(1), r_{i,t}(2), \ldots, r_{i,t}(M)]$ be M measurements taken for a flow i at a par-

ticular time of day t over some historical data set. Then for a flow i at time t, its acceptance probability or utility, $\Phi_{i,t}$, with respect to a given bandwidth allocation $x_{i,t}$ is defined as

$$\Phi_{i,t}[x_i(t)] = \frac{\#\ \text{measurements} \leqslant x_i(t)}{M} = \frac{1}{M}\{\sum_{k=1}^{M} I[r_{i,k}(t) \leqslant x_i(t)]\},$$

where $I[r_{i,k}(t) \leqslant x_i(t)]$ is the indicator that the measurement $r_{ij,k}(t)$ is less than or equal to $x_i(t)$.

3.B. Multipath Utility Max–Min Bandwidth Allocation

Although the utility max–min fair allocation problem has been well studied [23,24], it is limited to the context of single-path routing where the paths are given. In contrast, in our multipath utility max–min fair allocation problem, a flow can be routed over multiple paths and paths are variables in our problem rather than inputs. As shown in our evaluation results in Subsection 6.C, by considering the provisioning of circuits over multiple paths for a given IE pair, significantly greater headroom can be created to accommodate fluctuating traffic. By considering multipath bandwidth allocation, with the corresponding path diversity, our evaluation results in Subsection 6.C show that we are able to greatly improve the utilities (acceptance probabilities) that can be achieved, leading to much greater ability to accommodate actual traffic demand without the need for rerouting. To the best of our knowledge, we are the first to propose a solution to this open general multipath utility max–min fair allocation problem:

Definition 3 (Utility Max–Min Vector). *An allocation vector B is said to be utility max–min fair if it is feasible and each of its elements B_i cannot be increased without decreasing any other element B_k for which $\Phi_i(B_i) \leqslant \Phi_k(B_k)$.*

Definition 4 (Multipath Utility Max–Min Problem). *Given a network topology and a set of flows with their corresponding utility functions, a multipath utility max–min fair allocation problem achieves a utility max–min fair allocation vector B by choosing a set of paths P and determining the path bandwidth f for each flow in the network.*

In the above, we first define in Definition 3 what a utility max–min bandwidth allocation vector is. We then define in Definition 4 the corresponding definition for our multipath utility max–min problem. To solve this problem, we propose a variation of the water-filling algorithm with the help of a maximum concurrent flow (MCF) solver [38]. A MCF solver determines the maximum λ, the ratio of the offering traffic demand that can be routed in a network, and the corresponding routing paths P. If $\lambda < 1$, then it means there is no feasible routing to support the offered traffic demand completely.

Algorithm 1. Multipath Utility Max–Min Algorithm

1: Find the max common utility, φ, for Γ_{UNSAT} by binary searching domain [0, 1].
2: Compute the bandwidth requirement of each flow $i \in \Gamma_{\text{UNSAT}}$: $d_i = \Phi_i^{-1}(\varphi)$.
3: Compute the bandwidth requirement of each flow $j \in \Gamma_{\text{SAT}}$: $d_j = B_j$.
4: Query a MCF solver with demand d: $(\lambda, P) = \text{MCF}(d)$.
5: If $\lambda > 1$, increase φ; otherwise decrease φ.
6: Identify saturated flows.
7: For each flow $i \in \Gamma_{\text{UNSAT}}$:
8: Try to route an extra demand: $d'(i) = d(i) + \varepsilon$ and $d'(j) = d(j)\ \forall j \neq i$.
9: Query a MCF solver with demand d': $(\lambda', P') = \text{MCF}(d')$.
10: If $\lambda' > 1$, $\Gamma_{\text{newSAT}} = \Gamma_{\text{newSAT}} \cup i$.
11: Fixed utility for saturated flows.
12: For each flow $i \in \Gamma_{\text{newSAT}}$:
13: $\Gamma_{\text{SAT}} = \Gamma_{\text{SAT}} \cup i$; $\Gamma_{\text{unSAT}} = \Gamma_{\text{unSAT}} / i$; $B_i = \Phi_i^{-1}(\varphi)$.
14: Go back to Step 1 if $\Gamma_{\text{unSAT}} \neq \text{NULL}$.
15: Return bandwidth allocation B and routing paths P.

Algorithm 1 outlines the algorithm to achieve a utility max–min bandwidth allocation vector. The basic idea is to iteratively increase the utilities of the flows. If the width of a flow i cannot be increased without violating the max–min utility requirement, it is tagged as a saturated flow. Once a flow is tagged as saturated, it is added into the set Γ_{SAT}, and its utility φ_i and bandwidth allocation B_i are fixed in the later

iterations. However, the routing paths P_i are permitted to change and are recomputed in each iteration to achieve the maximum allocation vector. The algorithm repeats the water-filling process until all flows are saturated and their utilities cannot be increased further.

More specifically, the algorithm has three critical steps. The first is to find the maximum utility that can be achieved for all unsaturated flows by performing a binary search over the utility domain space [0, 1]. For a given utility φ, we verify its feasibility by using a MCF solver to test whether there exists a feasible routing P that can carry the corresponding required bandwidth of φ from the utility functions. The second step is to identify the bottleneck flows whose utility cannot be further increased. Different from the problem in the single-path formulation where the saturation of a flow can be determined by the residual capacity of its given path, in multipath formulation, a flow is truly saturated if and only if its bandwidth allocation cannot be increased by any routing path combination. Therefore, we again have to rely on a MCF solver to identify the saturation of a flow by testing whether we could route an extra demand ε for the flow. Finally, we fix the utility and bandwidth allocation of saturated flows at the end of each iteration. But notice that we allow the routing paths of all flows to be recomputed in Step 1 to achieve the maximum utility at each iteration without limiting the path selections.

4. Adaptive Rerouting over Circuits

4.A. Basic Idea

When a packet from flow i arrives at its ingress boundary router s_i, the boundary router first tries to send the packet by the direct circuit (s_i,d_i) by default. If there is no spare capacity left on the circuit from s_i to d_i, the boundary router i reroutes the packet to some other intermediate node k by the circuit (s_i,k). Upon receiving the packet, router k then again tries to send the packet by its direct circuit from k to d_i. If there is still no residual capacity left on circuit (k,j), the packet is rerouted to another intermediate router k'. The same process is repeated at router k'. Eventually, the packet would reach its destination, as long as there is spare capacity left from s_i to d_i along some path of circuits.

As shown in our evaluation results, this rerouting scheme greatly improves the network utilization and throughput on our circuit provisioning for several reasons. First, because our circuit provisioning establishes a full mesh of circuits on top of physical links, assuming there is traffic demand between every IE pair, there are N fan-out circuits at each router to handle a traffic burst, where N is the number of boundary routers. Thus, unless all fan-out circuits are saturated, the burst traffic can be rerouted to other intermediate routers. Second, at any intermediate router, there always exists a direct circuit to reroute the traffic. Therefore, with each reroute hop, the burst traffic has the opportunity to utilize the residual capacity from an intermediate node to its destination. In other words, in principal, the burst traffic would only be dropped when there is no residual circuit capacity left from any router to its destination.

4.B. Adaptive Load Balancing

Although there are many possible ways to reroute traffic by using different policies to select the next hop intermediate node at each router for rerouting traffic, we currently consider a local adaptive load-balancing algorithm. Specifically, when a router cannot route traffic through its direct circuit, it load balances the traffic to all other routers in proportion to the amount of residual capacity left on each circuit as the weighting factor. In doing so, a router evenly utilizes all outgoing residual capacity without overly saturating a particular circuit. In practice, the amount of residual capacity of a circuit could be either measured by its queue state, such as queue length, or a rate meter, such as a counter.

Another issue of our adaptive load-balancing routing is packet ordering caused by the dynamic routing path decision. The packet ordering on dynamic routing has been well studied and several solutions have been proposed [39]. In particular, a recently proposed solution [39] has shown that packet ordering can be maintained with a few kilobits of router state. This mechanism can be used in our adaptive load-balancing scheme.

Finally, since the adaptive load-balancing routing is a local algorithm and makes greedy decisions, it could cause routing loops along paths. Routing loops can be problematic. One issue is that a packet could suffer long delays or never reach its destination. Another issue is that routing loops simply waste circuit capacity, reducing network throughput and adding unnecessary load on intermediate routers. Therefore, it is critical to eliminate loops from our routing paths. One basic solution is to set the time-to-live (TTL) limit on packets, so that a packet would not be trapped in an infinite loop. Another innovative solution is to guarantee that a routing loop cannot be formed during our adaptive routing process by constructing a loop-free routing table, as described next.

4.C. Loop-Free Routing Table

As mentioned, the adaptive load-balancing rerouting algorithm proposed in Subsection 4.B could form routing loops and cause potential performance issues. In this subsection, we propose a solution to eliminate routing loops by constructing a loop-free routing table for each node. The basic idea of loop-free routing tables is to limit the path selections during the rerouting process so that a packet cannot select a node that has been visited before as its next intermediate node. At the same time, these loop-free routing tables should still preserve enough path diversities to allow the adaptive load-balancing rerouting algorithm to have sufficient dynamic routing ability to explore the available residual circuit capacities in a network.

To achieve the two requirements, for each IE pair, we compute its maximum acyclic graph [40] on top of the provisioned circuits. An acyclic graph assigns a direction for each circuit and guarantees that any routing path that follows these directed circuits are loop free. Then an acyclic graph is realized in a network by having a set of routing table entries associated with the source–destination prefixes for different IE pairs on each router. For example, if an acyclic graph has a directed edge from i to j for the IE pair from s to t, then we insert a table entry to node j on router i with the table index pair (s,t). Thus, when a packet from s to t arrives at router i, the packet can only be rerouted to one of the next nodes in its acyclic graph, such as j.

However, computing a maximum acyclic graph on an arbitrary graph has been shown to be NP-complete [40]. In our problem, however, the underlying graph is a fully connected mesh because our circuit provisioning has set up a circuit between each source and destination pair. In this special case, as indicated in [40], for any IE pair (s,t), its maximum acyclic graph can be found by inducing the acyclic subgraph G of any permutation order of nodes, where s is first in the order and t is last in the order. Specifically, we consider the following permutation of any flow (s,t):

$$s,\ s+1,\ s+2,\ \ldots,\ t-1,\ s-1,\ s-2,\ \ldots,\ t+2,\ t+1,\ t.$$

The above permutation order assumes $s=0$ and $t>s$. Since the nodes are symmetric, permutations for any s and t can be correspondingly defined by the appropriate shifting. This permutation produces an acyclic graph that can be represented as a ring, as shown in Fig. 3, and the traffic in the ring follows two simple routing rules. Consider that the ring is divided into two regions by the source and destination: the traffic within each region can only be traversed in either the clockwise or the counterclockwise direction. The second rule is that the traffic in the region starting from the right side of the source can jump to any node in the other region. Based on these two rules, the loop-free routing table can be constructed by a simple equation as shown in Fig. 4. Therefore, not only is the routing table construction time constant, but the routing table lookup time for a packet is also constant by checking the source, destination, and current node index.

5. Experimental Setup

We have extensively evaluated our proposed coarse-circuit-provisioning algorithm and our adaptive rerouting scheme on two separate real, large PoP-level backbone networks, namely, Abilene [1] and GEANT [2]. Both networks have been studied and discussed in the research literature, and their data sets, including network topology, routing information, and traffic measurements, are available in the public domain. Based on these available data sets, we implemented a flow-based trace-driven simula-

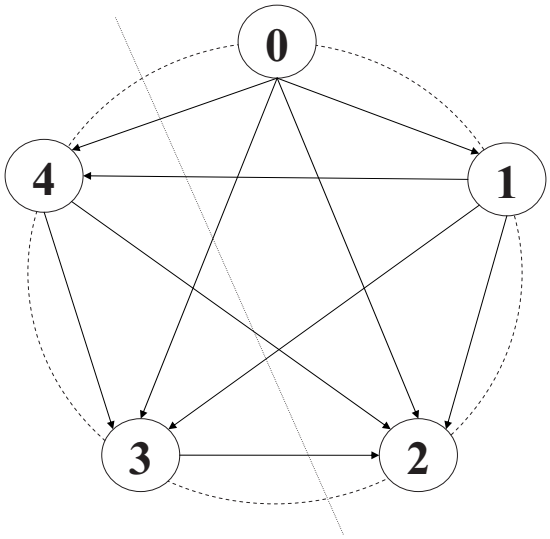

Fig. 3. Acyclic graph of flow from 0 to 2 in a 5 node network. The dotted line indicates the cut of two routing regions by source and destination.

src	dest	next hop (j)
0	1	$[1, 2]$
0	2	$[2, 2]$
0	4	$[4, 0]$
...		
4	0	$[0, 2]$
4	1	$[1, 2]$
4	2	$[2, 2]$
s	t	If $i \in [s, t]$, $j = [i + 1, s - 1]$. Otherwise, $j = [t, i - 1]$

Fig. 4. Routing table of node 3 in a 5 node network. The last entry shows the generic rule to construct the routing table for any node i for flow (s,t) in an N node network.

tor and conducted our evaluations. In the following, we first describe the two network topologies and their corresponding traffic matrices. Then we present our simulation setup in greater detail.

5.A. Network Topology

- Abilene: This is a public academic network in the U.S. with 11 nodes interconnected by OC192, 10 Gbits/s links.
- GEANT: This network connects with a variety of European research and education networks. The topology of GEANT has been slightly changed and has evolved in the past few years. Our experiments were based on the December 2004 snapshot available at [41]. According to the snapshot, the network consists of 23 nodes and 74 links varied from 155 Mbits/s to 10 Gbits/s.

5.B. Traffic Matrices

Traffic matrices are used in our experiments both for deriving circuit configurations using our proposed algorithms as well as for creating simulation traffic for our performance evaluations. A traffic matrix consists of the demand rate (kbits/s) of every IE pair within a certain time interval (e.g., 5 min for Abilene and 15 min for GEANT). Therefore, these traffic matrices provide a snapshot of real total demand offerings between each IE pair in a network at different time instances. For both networks, we used the real traffic matrices provided from a third party [42]. The traffic matrix data sets for the Abilene network are available at [43], and the traffic matrix data sets of the GEANT network are available at [44].

In brief, these traffic matrices are derived based on the flow information collected from key locations of a network by traffic monitors, such as Netflow [45]. Then the flow information is transformed into the demand rate of each IE pair in a traffic matrix based on the routing information in the network. We collected the traffic

matrices in each network for an extended period of time to represent the historical traffic measurements and simulation traffic load. Detailed information on the traffic matrices used on the two networks are summarized in Table 1.

5.C. Simulation

To facilitate evaluation at scale, we implemented a flow-based simulator. The simulator is not packet-based and is intended to simulate only the steady-state network behavior. Thus the simulator is not subject to notions of propagation, transmission, or queuing delay. The inputs to the simulator are a network topology annotated with link capacities and a traffic matrix. According to the demand rates of IE flows from the given traffic matrix, we route the demand of flows through the network and keep track of the bandwidth consumption along their routing paths. Then we determine the amount of demand from each IE flow that can actually go through the network by computing how much demand has to be dropped from the network for each IE flow. Specifically, since our experiments compare both packet routing [e.g., open shortest path first (OSPF) and equal cost multiple path (ECMP)] and circuit switching, the demand of IE flows is considered to be dropped when the aggregated demand from IE flows exceeds the bandwidth of a link (under a packet routing) or exceeds the allocated capacity of a circuit (under a circuit switching). Therefore, for each IE flow, we are able to compute its drop rate as the amount of demand dropped from the flow as a percentage of its offering demand in the traffic matrix. For simplicity, we can think of the drop rate of an IE flow as the opposite of its throughput under the given network offering demand and routing algorithm.

In Section 6, we compare different routing algorithms by showing the IE flow drop rate under varied traffic offerings, and we use the traffic matrices collected from [43,44] as the different traffic demand inputs. The exact traffic matrices we used in the experiments as simulated traffic offerings are the ones observed across a week (Monday to Friday) for both networks (Abilene: April 22–26, 2004; GEANT: April 11–15, 2004). Since we have a traffic matrix every 5 min for Abilene and every 15 min for GEANT, we reported the IE flow drop rate over 1,440 and 288 different traffic demand inputs for Abilene and GEANT, respectively. With consistent improvements over this wide range of traffic profiles, it provides confidence regarding what our new proposed approach can achieve.

6. Experiments

We first evaluate the performance of COPLAR in Subsection 6.A. We then compare COPLAR with two conventional packet-routing algorithms (OSPF, ECMP) and a circuit-switching (two-phase) routing algorithm in Subsection 6.B. Finally, we investigate the advantage of multipath circuit provisioning formulation for COPLAR in Subsection 6.C and the robustness of COPLAR under unexpected traffic in Subsection 6.D.

6.A. Evaluation of COPLAR

This subsection evaluates the performance of COPLAR using the real data traffic trace across a 5 day period (Monday to Friday). COPLAR reconfigured its circuits at each hour based on the historical traffic measurements. In particular, we compare the results of the COPLAR and the COPLAR without rerouting scheme. COPLAR without rerouting is referred to as COPLAR-NR. For both COPLAR and COPLAR-NR, with a given traffic matrix, we measured its performance by computing a drop rate, which is the total demand dropped as a percentage of the total offered load from all IE flows. Since the time granularity of traffic matrices in Abilene is 5 min, there are a total of 1,440 consecutive data points over the 5 day simulation period. Similarly, the GEANT network has 288 data points from its 15 min traffic matrices.

Table 1. Traffic Matrices Information for Abilene and GEANT

Network	Collection Period	Circuit Provision History Traffic Matrix	Simulated Traffic Matrix	Time Interval
Abilene	03/01/04–04/26/04	03/01/04–04/21/04	04/22/04–04/26/04	5 minutes
GEANT	01/01/05–04/15/05	01/01/05–04/10/05	04/11/05–04/15/05	15 minutes

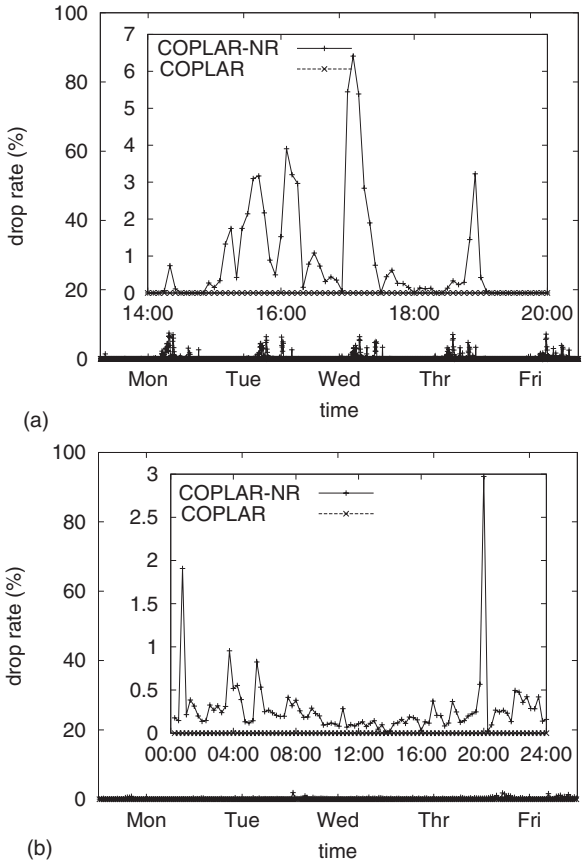

Fig. 5. Traffic drop rate across a week under COPLAR and COPLAR-NR. The inner graphs enlarge the results of (a) Abilene from 2 p.m. to 8 p.m. on Wednesday and the results of (b) GEANT on Wednesday.

As we know, current networks are underutilized. To demonstrate that our circuit provisioning can achieve low drop rates in a network with high utilization, we normalized our simulated traffic matrices by scaling their offering load by some factor, such that at least one traffic matrix had traffic drops under a single shortest path routing (OSPF [29]). The scale factor for Abilene and GEANT were roughly equal to 4 and 2, respectively. The drop rates of each of the normalized traffic matrices across the 5 day simulation are reported in Fig. 5. As shown in the figure, even without our rerouting scheme, COPLAR-NR achieves relatively low drop rates over all traffic matrices. In particular, the inner graphs of Fig. 5 enlarge the drop rates on a particular day (Wednesday) and show that the maximum drop rates of COPLAR-NR are only 7% and 3% in Abilene and GEANT, respectively. Therefore, our provisioning algorithm did effectively allocate bandwidth and minimize the drop rate even in a congested network. Furthermore, by combining circuit configurations with rerouting, COPLAR is able to achieve 0% drop rates for all simulated traffic by utilizing the spare capacity of circuits.

6.B. Routing Comparison

We then compare our proposed solutions with two commonly used packet-switching algorithms, OSPF and ECMP, and a recently proposed static circuit-routing algorithm, two-phase [8]. OSPF [29] is an implementation of Dijkstra's single shortest path algorithm [46]. OSPF is commonly used as a routing algorithm for packet switching. Specifically, we computed the OSPF routes based on the link weights given from the network topology data sets. ECMP [30] is a Cisco router implementation of OSPF with the ability to load balance traffic across all the shortest paths that have equal cost (sum of link weights along a path). Finally, two-phase is a novel routing algorithm that aims to minimize the maximum link utilization under all admissible traffic bounded by the access capacity of boundary routers. Because the routing is inde-

pendent of the actual traffic offering, two-phase can provision the required bandwidth in advance and configure static circuits at once.

To demonstrate that COPLAR utilizes network capacity effectively, we compare the drop rate under varied network utilizations by using the normalized traffic matrices scaled by a factor of 0 to 3 as the input offering load. Under a given traffic scale factor, we again computed the drop rates of all simulated traffic matrices using each of the algorithms. Instead of showing the drop rate of each simulated traffic matrix, we present the summary results under each traffic scale factor in Figs. 6 and 7. Figure 6 plots the mean drop rates of all simulated traffic matrices at each traffic scale factor. Figure 7 plots the 90th percentile number of the drop rates among all simulated traffic matrices at each traffic scale factor. Since both figures have a similar trend with slightly higher drop rates for 90th percentile numbers, in the following, we discuss the results using the numbers in Fig. 7.

Clearly, all algorithms have higher drop rates as the traffic scale factor is increased. But two-phase routing has the highest drop rates because it optimizes circuits for the worst case in all admissible traffic matrices, which is too pessimistic. On the other hand, although COPLAR-NR only routes traffic by direct circuits, it is still able to achieve much lower drop rates than two-phase. As shown in Fig. 7, in both networks, two-phase starts dropping traffic from a factor of 0.5, whereas COPLAR-NR remains at a nearly 0% drop rate until a factor of 1.0. That means COPLAR-NR is able to carry as much as twice the traffic load as two-phase. Also, we found that, even at a factor of 3.0, COPLAR-NR has a drop rate reduction over two-phase by 55.85% (reduced from 55.70% to 24.59%) and 47.04% (reduced from 22.62% to 11.98%) in Abilene and GEANT, respectively.

In both networks, OSPF starts having drops after a factor of 1.0 because our normalized traffic already scaled the offering load based on OSPF to create congestions. ECMP has lower drop rates than OSPF because it has more path diversities and load

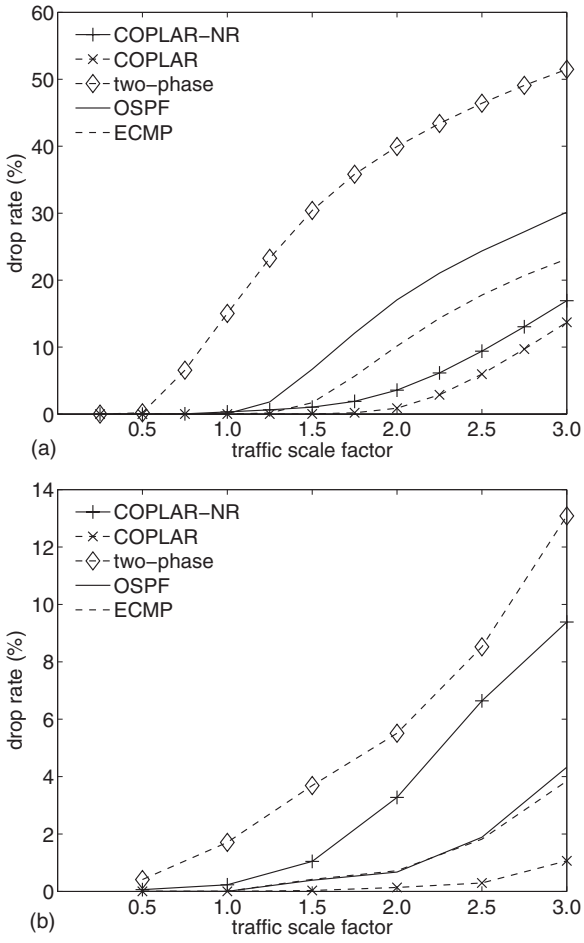

Fig. 6. Mean drop rate comparison of COPLAR with other routing algorithms as the traffic is scaled by a factor of 0 to 3.

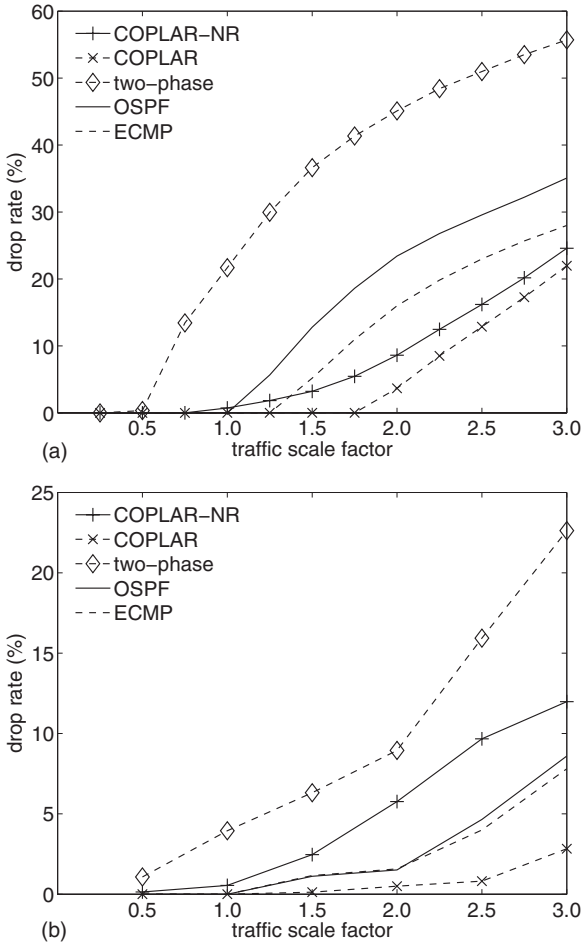

Fig. 7. 90th percentile drop rate comparison of COPLAR with other routing algorithms as the traffic is scaled by a factor of 0 to 3.

balances traffic on multiple paths. This strategy is particularly useful when a network has certain bottleneck paths, such as Abilene. This is because in Abilene, under OSPF, most of the traffic routes through the links between Indianapolis and Denver and causes bottlenecks with higher link utilizations. A multipath routing algorithm such as ECMP minimizes congestion by routing traffic through Houston. As a result, ECMP has consistently lower drop rates than OSPF and does not have any drop in Abilene until a factor of 1.25. Compared with OSPF and ECMP, in GEANT, COPLAR-NR has higher drop rates because it cannot share the residual capacity on circuits among flows. However, in Abilene, COPLAR-NR actually has a lower drop rate than OSPF and ECMP when the traffic scale is larger than 1.5. This is because our bandwidth allocation algorithm is able to utilize link capacities even more effectively than ECMP by carefully chosen circuit configurations, whereas OSPF and ECMP only have very limited routing path selections.

On the other hand, COPLAR has consistently lower drop rates than OSPF and ECMP in both networks. Again, this is because COPLAR has the ability to fully explore path diversity and perform adaptive routing on the fly. Specifically, COPLAR maintains a nearly 0% drop until a factor of 1.75 and 1.5 in Abilene and GEANT, respectively. In other words, COPLAR can support 75% and 50% more traffic load than OSPF in the two networks. Even compared with ECMP, under much high traffic load at a factor of 3.0, COPLAR has a drop rate reduction over ECMP by 21.51% (reduced from 27.99% to 21.97%) in Abilene and 63.75% (reduced from 7.80% to 2.83%) in GEANT. COPLAR is able to achieve a greater drop rate reduction in GEANT because there exists more residual capacity as indicated by the lower drop rate in GEANT than in Abilene.

6.C. Single-Path versus Multipath Circuit Provisioning

Here, we analyze the performance improvement from solving our circuit-provisioning algorithm as a multipath problem rather than as a single-path problem. Because cir-

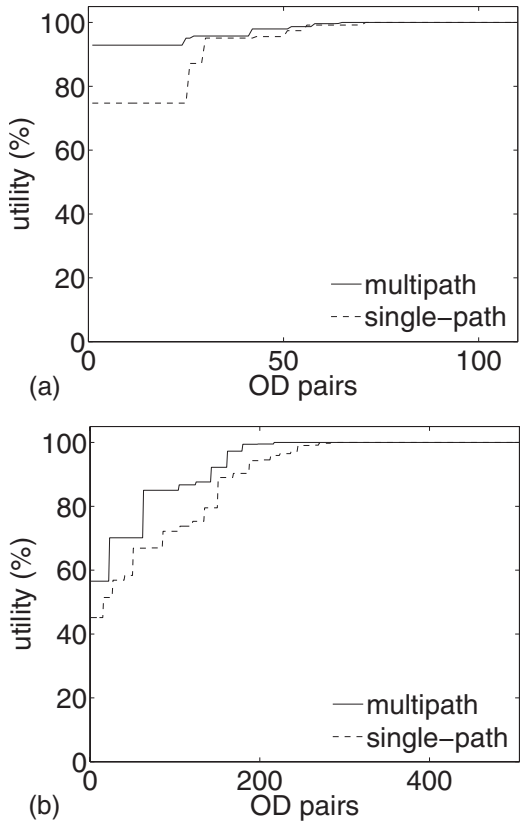

Fig. 8. Utility achieved for each IE flow under single-path and multipath utility max–min problem formulation.

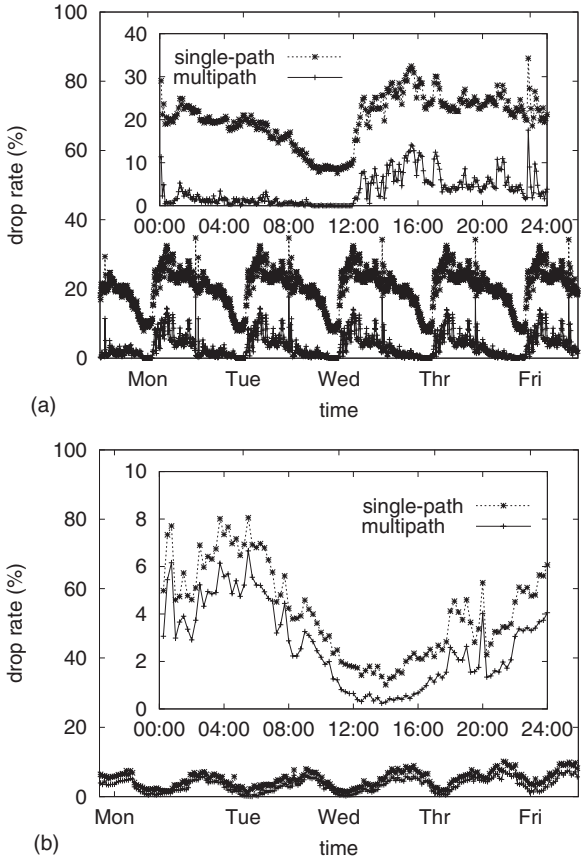

Fig. 9. Traffic drop rate across a week under single-path and multipath circuit configurations. The inner graphs enlarge the results on Wednesday for both networks.

cuit provisioning plays a more important role when a network has higher utilization, and link capacity is more scarce, we present the circuit-provisioning results under the normalized traffic matrices with a scale factor of 2 in this subsection. The multipath solution was computed by the algorithm outlined in Algorithm 1, while the single-path solution was derived by the traditional single-path water-filling algorithm [23,24].

First, with a given network topology and utility functions of traffic distribution, we compare the utility (acceptance probability) that can be achieved under a multipath solution and a single-path solution. As an example, we present the bandwidth allocation results computed at the peak hour of 5 p.m. on Wednesday for both networks. Figure 8 plots the utilities of each IE flow, and the IE flows are sorted by their utilities in the figure. As shown in the figure, the multipath algorithm achieves strictly higher utility for all IE pairs in both networks. In particular, compared with the single-path algorithm, our multipath algorithm maximizes the minimum utility from 74.74% to 92.90% in Abilene and from 45.15% to 56.54% in GEANT. Furthermore, the number of IE pairs with 100% utility also increases from 40 to 46 in Abilene and 223 to 290 in GEANT.

We then show the effect of multipath formulation on circuit configuration in practice. Figure 9 plots the drop rates of single-path and multipath solutions. Again, we present the drop rates across 5 days of simulated traffic. Notice that the multipath results shown in Fig. 9 are higher than the drop rate of COPLAR-NR in Fig. 5 because Fig. 9 has twice as much traffic as Fig. 5. As shown in the figure, the multipath solution significantly reduces drop rates at all time instances in both networks, especially in the Abilene network where single-path routing suffers higher drop rates because of the existence of bottleneck links. For example, at 4 a.m. on Wednesday in Abilene, the multipath circuit configuration reduces the drop rate of the single-path solution by 91.30% from 20.12% to 1.75%. Across all the simulated traffic matrices, the multipath

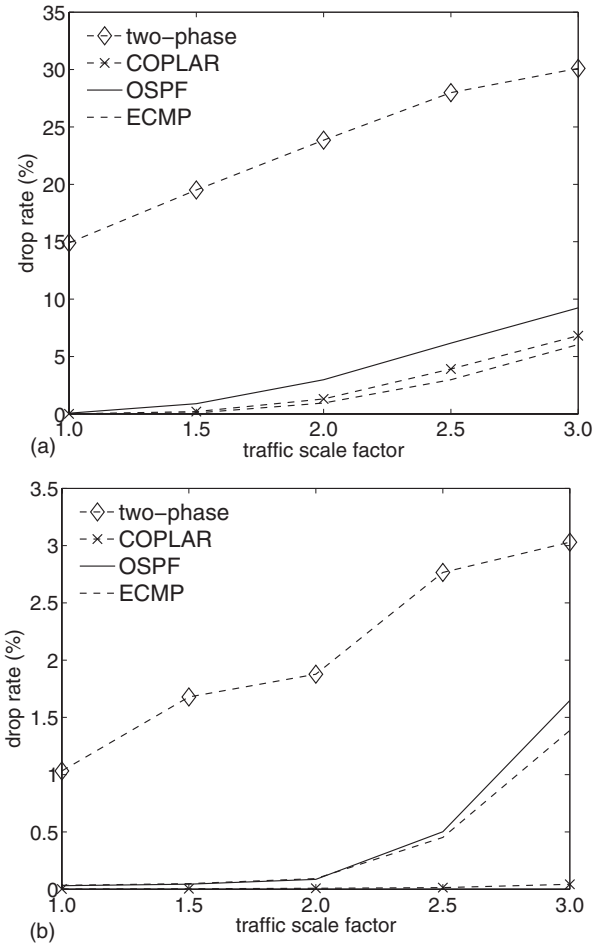

Fig. 10. Mean drop rate under hot-spot traffic matrices as the demand of hot-spot flows are scaled by a factor of 1 to 3.

solution reduces the drop rate of the single-path solution by 46.69% to 100% in Abilene and 15.88% to 98.51% in GEANT. Therefore, the results in Figs. 8 and 9 strongly indicate the importance and advantage of multipath problem formulation in our circuit-provisioning approach.

6.D. Evaluation of Robustness

Finally, we show the robustness of COPLAR by observing the performance when unexpected traffic occurs. We generated unexpected traffic by a hot-spot model that aims to simulate the scenario where there is a sudden burst for the traffic to certain destinations. Specifically, a hot-spot traffic matrix was generated as follows. First, we randomly picked a traffic matrix from our normalized traffic matrices. Then in the traffic matrix, we randomly selected a few hot-spot nodes and scaled all the flows to these nodes by a factor of 1.0 to 3.0. In the simulation, we picked approximately 25% of nodes as hot spots. That is, three hot-spot nodes in Abilene and five hot-spot nodes in GEANT. For each scaling factor, we randomly generated 100 testing hot-spot traffic matrices, and we present the mean and 90th percentile drop rate on those matrices in Figs. 10 and 11, respectively.

As shown in the figures, COPLAR achieves significantly lower drop rates than two-phase and OSPF in both networks. For example, in Fig. 11(a), although COPLAR still has a 0% drop at a factor of 1.5, two-phase and OSPF already suffer a drop rate of 27.73% and 3.81%, respectively. COPLAR also has a consistently lower drop rate over all scale factors. Even with the burst traffic load at a factor of 3.0, COPLAR reduces the drop rate of OSPF by 19.34% (from 21.92% to 17.68%) and reduces the drop rate of two-phase by 56.97% (41.09% to 17.68%).

Although the drop rate of COPLAR is close to ECMP in Abilene, COPLAR maintains a nearly 0% drop under all traffic factors in GEANT. On the other hand, even ECMP

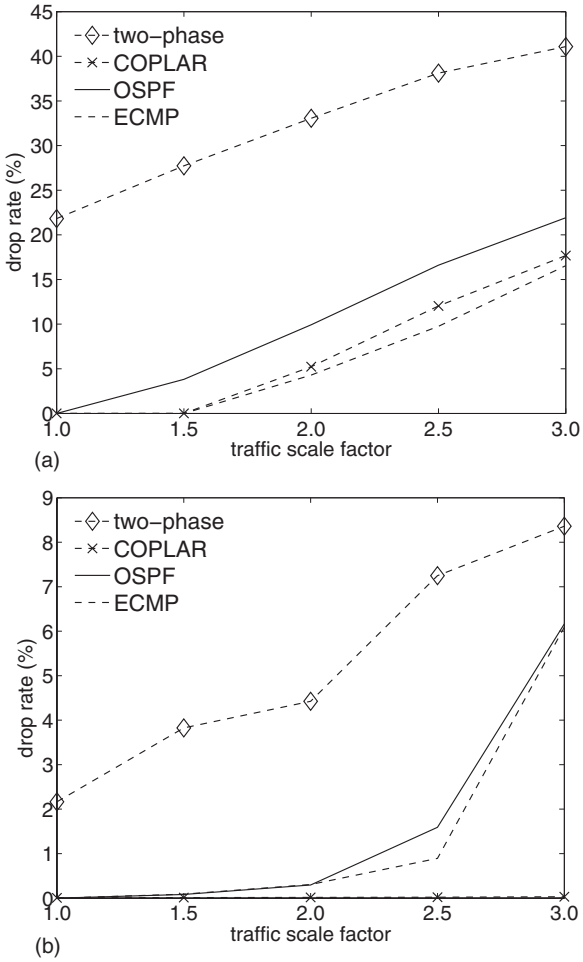

Fig. 11. 90th percentile drop rate under hot-spot traffic matrices as the demand of hot-spot flows are scaled by a factor of 1 to 3.

has up to a 6% drop rate at a factor of 3.0. COPLAR is able to accommodate the burst traffic because of its ability to explore much greater path diversity than ECMP, which only considers the paths with equal cost. Also, in GEANT, there is more residual capacity left under normalized traffic demand as indicated by the much smaller mean and 90th drop rate compared with Abilene. Therefore, although the burst traffic can easily exceed the allocated bandwidth for its dedicated circuit, our rerouting scheme is still able to find residual capacity along other alternated circuits to reroute the burst traffic and minimize impacts.

7. Conclusion

In this paper, we have proposed a new solution for optical networking based on the paradigm of optical circuit switching by default and packet rerouting over circuits with spare capacity when necessary. We considered the idea of provisioning long-duration quasi-static optical circuits between edge routers at the boundary of the network to carry the traffic by default. When the provisioned circuit is inadequate, excess traffic demand is rerouted through circuits with spare capacity. In particular, by adaptively load balancing across circuits with spare capacity, excess traffic is routed to its final destination without the need to create circuits on the fly. Our approach was extensively evaluated with actual traffic traces on two separate real, large ISP PoP-level topologies, Abilene [1] and GEANT [2]. Using our proposed solution, our results show that only a very small amount of excess traffic needs to be rerouted even during peak traffic hours when the circuit configurations are carefully chosen. With adaptive rerouting, our approach was able to accommodate this excess traffic. We also demonstrate that our adaptive load-balancing approach is robust to sudden unexpected traffic changes by demonstrating its ability to reroute traffic under a number of hot-spot scenarios.

References and Notes

1. V. Chan, "Near-term future of the optical network in question?" IEEE J. Sel. Areas Commun. **25**(9, Part Supplement), 1–2 (2007).
2. M. Yoo, C. Qiao, and S. Dixit, "Optical burst switching for service differentiation in the next-generation optical Internet," IEEE Commun. Mag. **39**(2), 98–104 (2001).
3. G. Qiao and M. Yoo, "Optical burst switching (OBS)—a new paradigm for an optical Internet," J. High Speed Networks **8**(1/1999), 69–84 (1999).
4. P. Molinero-Fernandez and N. McKeown, "TCP switching: exposing circuits to IP," in *Hot Interconnects 9* (IEEE, 2001), pp. 43–48.
5. P. Molinero-Fernandez, "Circuit switching in the Internet," Ph.D. thesis (Stanford University, 2003).
6. R. Zhang-Shen and N. McKeown, "Designing a predictable Internet backbone network," in *HotNets III* (Association for Computing Machinery, 2004).
7. M. Kodialam, T. V. Lakshman, and S. Sengupta, "Efficient and robust routing of highly variable traffic," in *HotNets III* (Association for Computing Machinery, 2004), pp. 15–21.
8. 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 *25th IEEE International Conference on Computer Communications* (IEEE, 2006), pp. 1–12.
9. V. W. S. Chan, G. Weichenberg, and M. Medard, "Optical flow switching," in *Workshop on Optical Burst Switching* (IEEE, 2006), pp. 1–8.
10. G. Weichenberg, V. W. S. Chan, and M. Medard, "On the capacity of optical networks: a framework for comparing different transport architectures," IEEE J. Sel. Areas Commun. **25**, 84–101 (2007).
11. A. Banerjee, J. Drake, J. Lang, B. Turner, D. Awduche, L. Berger, K. Kompella, and Y. Rekhter, "Generalized multiprotocol label switching: an overview of signaling enhancements and recovery techniques," in IEEE Commun. Mag. **39**(1), 144–150 (2001).
12. S. J. B. Yoo, "Optical-label switching, MPLS, MPLambdaS, and GMPLS," Opt. Networks Mag. **4**(3), 17–31 (2003).
13. G. Bernstein, B. Rajagopalan, and D. Spears, "OIF UNI 1.0—controlling optical networks," White paper (Optical Internetworking Forum, 2001).
14. COPLAR is pronounced the same as the word "copular," which is the adjective form of the noun "copula," meaning "something that connects or links together."
15. M. Roughan, A. Greenberg, C. Kalmanek, M. Rumsewicz, J. Yates, and Y. Zhang, "Experience in measuring Internet backbone traffic variability: models, metrics, measurements and meaning," in *International Teletraffic Congress (ITC)* (2003), pp. 379–388.
16. A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot, "Traffic matrix estimation: existing techniques and new directions," Comput. Commun. Rev. **32**, 161–174 (2002).

17. D. Banerjee and B. Mukherjee, "Wavelength-routed optical networks: linear formulation, resource budgeting tradeoffs, and a reconfiguration study," IEEE/ACM Trans. Netw. **8**, 598–607 (2000).
18. B. Ramamurthy and A. Ramakrishnan, "Virtual topology reconfiguration of wavelength-routed optical WDM networks," in *IEEE Global Telecommunications Conference* (IEEE, 2000), pp. 1269–1275.
19. M. Tornatore, G. Maier, and A. Pattavina, "WDM network optimization by ILP based on source formulation," in *Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies* (IEEE, 2002), pp. 1813–1821.
20. F. Ricciato, S. Salsano, A. Belmonte, and M. Listanti, "Off-line configuration of a MPLS over WDM network under time-varying offered traffic," in *Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies* (IEEE, 2002), pp. 57–65.
21. D. Applegate and E. Cohen, "Making intra-domain routing robust to changing and uncertain traffic demands: understanding fundamental tradeoffs," in *ACM SIGCOMM* (Association for Computing Machinery, 2003), pp. 313–324.
22. D. Bertsekas and R. Gallager, *Data Networks* (Prentice Hall, 1987).
23. Z. Cao and E. W. Zegura, "Utility max–min: an application-oriented bandwidth allocation scheme," in *Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies* (IEEE, 1999), pp. 793–801.
24. D. Rubenstein, J. Kurose, and D. Towsley, "The impact of multicast layering on network fairness," IEEE/ACM Trans. Netw. **10**, 169–182 (2002).
25. B. Radunovic and J. Y. Le Boudec, "A unified framework for max–min and min–max fairness with applications," IEEE/ACM Trans. Netw. **15**, 1073–1083 (2007).
26. M. Allalouf and Y. Shavitt, "Centralized and distributd approximation algorithms for routing and weighted max-min fair bandwidth allocation," in *2005 Workshop on High Performance Switching and Routing* (IEEE, 2005), pp. 306–311.
27. Advanced networking for leading-edge research and education, http://abilene.internet2.edu.
28. Intel-Dante monitoring project, http://www.geant.net/server/show/nav.117.
29. J. Moy, OSPF version 2, March 1994, www.ieft.org/rfc/rfc2328.txt.
30. C. Hopps, "Analysis of an equal-cost multi-path algorithm," Request for Comments 2992 (Network Working Group, 2000).
31. E. Rose, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture, Request for Comments 3031 (Network Working Group, 2001).
32. L. Xu, H. G. Perros, and G. Rouskas, "Techniques for optical packet switching and optical burst switching," IEEE Commun. Mag. **39**(1), 136–142 (2001).
33. S. Yao, B. Mukherjee, S. J. B. Yoo, and S. Dixit, "A unified study of contention-resolution schemes in optical packet-switched networks," J. Lightwave Technol. **21**, 672–683 (2003).
34. S. J. B. Yoo, F. Xue, Y. Bansal, J. Taylor, Z. Pan, J. Cao, M. Jeon, T. Nady, G. Goncher, K. Boyer, K. Okamoto, S. Kamei, and V. Akella, "High-performance optical-label switching packet routers and smart edge routers for the next generation internet," IEEE J. Sel. Areas Commun. **21**, 1041–1051 (2003).
35. S. Kandula, D. Katabi, B. Davie, and A. Charny, "Walking the tightrope: responsive yet stable traffic engineering," in *ACM SIGCOMM* (Association for Computing Machinery, 2005), pp. 253–264.
36. A. Elwalid, C. Jin, S. Low, and I. Widjaja, "MATE: MPLS adaptive traffic engineering," in *Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies* (IEEE, 2001), pp.1300–1309.
37. S. Fischer, N. Kammenhuber, and A. Feldmann, "REPLEX—dynamic traffic engineering based on Wardrop routing policies," in *ACM CoNEXT* (Association for Computing Machinery, 2006), pp. 1–12.
38. G. Karakostas, "Faster approximation schemes for fractional multicommodity flow problems," in *Thirteenth Accnual ACM-SIAM Symposium on Discrete Algorithms* (Association for Computing Machinery, 2002), pp. 166–173.
39. S. Kandula, D. Katabi, S. Sinha, and A. Berger, "Dynamic load balancing without packet reordering," Comput. Commun. Rev. **37**, 51–62 (2007).
40. R. Hassin and S. Rubinstein, "Approximations for the maximum acyclic subgraph problem," Inf. Process. Lett. **51**, 133–140 (1994).
41. Available at http://www.geant.net/upload/pdf/GEANT_Topology_12-2004.pdf.
42. TOTEM, a toolbox for traffic engineering methods, February 2005, http://totem.info.ucl.ac.be.
43. Y. Zhang, Abilene traffic matrices, www.cs.utexas.edu/~yzhang/research/AbileneTM.
44. TOTEM GEANT traffic matrices, totem.info.ucl.ac.be/dataset.html.
45. Cisco IOS Netflow, http://www.cisco.com/warp/public/732/Tech/netflow/.
46. E. Dijkstra, "A note on two problems in connexion with graphs," Numerische Matematik **271**, 1–269 (1959).