Rehoming Edge Links for Better Traffic Engineering

Eric Keller
University of Pennsylvania
kellere@seas.upenn.edu

Michael Schapira
Hebrew University of Jerusalem
schapiram@huji.ac.il

Jennifer Rexford
Princeton University
jrex@cs.princeton.edu

ABSTRACT

Traditional traffic engineering adapts the routing of traffic *within* the network to maximize performance. We propose a new approach that also adaptively changes where traffic *enters* and *leaves* the network—changing the "traffic matrix", and not just the intradomain routing configuration. Our approach does *not* affect traffic patterns and BGP routes seen in neighboring networks, unlike conventional inter-domain traffic engineering where changes in BGP policies shift traffic and routes from one edge link to another. Instead, we capitalize on recent innovations in edge-link migration that enable seamless rehoming of an edge link to a different internal router in an ISP backbone network—completely transparent to the router in the neighboring domain. We present an optimization framework for *traffic engineering with migration* and develop algorithms that determine which edge links should migrate, where they should go, and how often they should move. Our experiments with Internet2 traffic and topology data show that edge-link migration allows the network to carry 18.8% more traffic (at the same level of performance) over optimizing routing alone.

Categories and Subject Descriptors

C.2.6 [**Computer Communication Networks**]: Internetworking; C.2.1 [**Computer Communication Networks**]: Network Architecture and Design

General Terms

Design, Experimentation, Management, Measurement

Keywords

Internet, architecture, routing, traffic engineering, migration

1. INTRODUCTION

The rapid growth of online services, from video streaming to 3D games and virtual worlds, is placing tremendous demands on the underlying networks. ISP backbone networks carry more traffic than ever. To address these challenges, network operators do *traffic engineering* (TE). Traditionally, traffic engineering involves tuning routing-protocol parameters to control how traffic is routed across the network, to optimize performance and use network resources effectively. Thus, today's traffic engineering adapts routing *within* the network for a given *traffic matrix*, *i.e.*, the volume of traffic between *fixed* traffic ingress and egress points. Traditional traffic engineering assumes that the topology is fixed and cannot be changed. Recent innovations challenge this approach.

A recently proposed mechanism—"*router grafting*" [8]—allows an ISP to dynamically rehome its ends of links to neighboring networks. For example, an ISP could move a customer that connects in New York to a different router in nearby New Jersey, transparently to the customer. With this capability, traffic engineering can go beyond adapting the routing protocol to control where traffic enters and exits the network. In effect, an ISP now has the power to *change the traffic matrix* without disrupting its neighbors. This flexibility gives rise to new possibilities in traffic engineering. Dynamically relocating traffic end-points can redirect traffic to decrease the traffic traversing a congested bottleneck, or capitalize on unused bandwidth.

In this paper, we introduce *traffic engineering with migration*, and present a framework for determining which edge links should migrate, and where they should migrate to. We present and analyze traffic engineering with migration in the context of edge-link migration via router grafting in backbone ISP networks.

1.1 Edge-Link Migration via Router Grafting

An ISP network connects to neighboring networks (customers, peers, or providers) at its perimeter. To establish a link to another network, the ISP selects one of its routers to connect to the adjacent network. Traditionally, the link remains fixed unless there is significant reason for change. This is because changing to a different internal router in real time can be extremely disruptive to the Border Gateway Protocol (BGP) session with the neighboring network, requiring significant coordination such as scheduling a maintenance window. During the transition period, data packets may be lost or delivered out of order, and routers throughout the Internet receive additional BGP update messages.

New mechanisms for rehoming links make the change transparent [1, 8]. Router grafting [8] enables an ISP to move its end of the link without disrupting user performance and without coordination with the neighboring network; the earlier RouterFarm [1] does so with slight downtime. Router grafting rehomes the layer-three link (through signaling in the programmable transport network), migrates the local end-point of the TCP connection to the neighbor's router, and transparently transfers the routing-protocol state to a different internal router. Router grafting has no impact on the neighboring network—the neighbor is not aware that grafting has happened, and sees no change in where traffic enters or leaves its own routers. This is in sharp contrast to traditional inter-domain traffic engineering, where an ISP changes its BGP policies to shift traffic from one edge link to another—triggering both BGP update messages and

changes in where traffic enters or leaves neighboring networks [10, 12, 14, 15].

Router grafting enables an ISP to migrate a link within a few seconds without disruption, allowing network operators to change the ingress and egress points for traffic in real time. The overhead of router grafting is relatively low. Grafting involves the export of state from one router, the transference of state, and the import of state at another router. Changing the network topology requires some routers to repeat the route-selection process, leading to a temporary increase in CPU load. In addition, some routers may change their routing decisions, leading to a temporary increase in BGP update messages. These overheads are short-lived, and do not disrupt the flow of data traffic. As such, network operators can afford to make periodic adjustments to where they terminate the links to neighboring networks.

1.2 Traffic Engineering with Migration

We develop techniques to determine which edge-links should migrate, to where, and how often. We first present a formal framework for traffic engineering with edge-link migration. We show that finding the optimal solution, or even a reasonable approximation of the optimal solution, in this new traffic-engineering setting is computationally intractable. We then present two relatively simple heuristics for traffic engineering with edge-link migration and show that they offer significant performance improvements in practice. Our experiments with Internet2 traffic and topology data show that migration would enable the network to carry 18.8% more traffic at the same level of performance. Importantly, we can achieve close to this level of improvement without frequently re-optimizing the topology—performing this re-optimization every 12 hours still sees a nearly 15% improvement. Similarly, only a small fraction of links need to be migrated in each interval. Migrating about 10% of the links results in the full improvement, but migrating less than 5% of the links still achieves 95% of the benefits.

Organization. After a brief review of traditional traffic engineering, we introduce traffic engineering with migration in Section 2. Section 3 shows that computing an optimal solution is hard, and presents two heuristics for traffic engineering with migration. Section 4 presents our experimental evaluation. We wrap up with a presentation of related work in Section 5, and conclusion in Section 6.

2. TRAFFIC ENGINEERING MODEL

2.1 Traffic Engineering Today

In traditional traffic engineering, the network is represented by a directed graph $G = (V, E)$, where the vertex set V represents routers, and the edge set E represents the links. Every edge $e \in E$ has capacity $c_e > 0$. We are also given a *traffic matrix* $D = \{d_{ij}\}_{i,j \in V}$, where entry $d_{ij} \geq 0$ is the amount of traffic that vertex i wishes to send vertex j. The goal is to distribute flow across the paths from i to j to minimizing total link usage (TLU).

TLU minimization reflects a common goal in ISP networks [5]. Each link e has a "cost" that reflects its level of congestion, where lightly-loaded links are "cheap" and links become exponentially more "expensive" as the link becomes heavily loaded. The *cost function* ϕ_e specifies the cost as a function of f_e (the total flow traversing the edge) and c_e

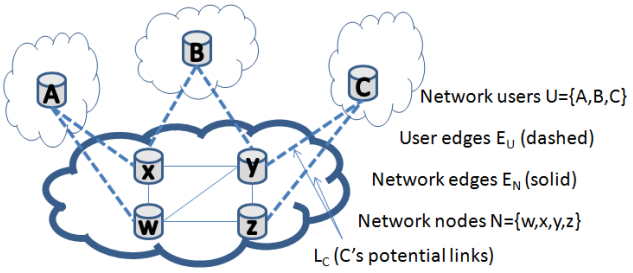

Figure 1: Network model for traffic engineering with migration.

(the edge capacity). Every ϕ_e is a piecewise linear, strictly increasing and convex function. We use the following cost function which was derived in [5] based on their evaluation:

$$\phi_e(f_e, c_e) = \begin{cases} f_e & 0 \leq \frac{f_e}{c_e} < \frac{1}{3} \\ 3f_e - \frac{2}{3}c_e & \frac{1}{3} \leq \frac{f_e}{c_e} < \frac{2}{3} \\ 10f_e - \frac{16}{3}c_e & \frac{2}{3} \leq \frac{f_e}{c_e} < \frac{9}{10} \\ 70f_e - \frac{178}{3}c_e & \frac{9}{10} \leq \frac{f_e}{c_e} < 1 \\ 500f_e - \frac{1468}{3}c_e & 1 \leq \frac{f_e}{c_e} < \frac{11}{10} \\ 5000f_e - \frac{16318}{3}c_e & \frac{11}{10} \leq \frac{f_e}{c_e} < \infty \end{cases}$$

The goal is to distribute the *entire* demand between every pair of vertices in a manner that minimizes the sum of all link costs (i.e., $\Sigma_{e \in E}\ \phi(f_e, c_e)$). (Observe that the flow along an edge can exceed the edge's capacity.) TLU minimization can be formulated as *minimum-cost multicommodity flow* and is thus computable using existing algorithms for computing multicommodity flows. Realizing this objective in practice can be done via MPLS and a management system that solves the optimization problem and installs the resulting paths. Network operators often take the indirect approach of tuning Interior Gateway Protocol (IGP) weights to approximate the optimal distribution of the traffic [5].

2.2 Migration-Aware Traffic Engineering

We now extend the traffic-engineering model in Section 2.1 to incorporate migration. Table 1 summarizes the notation.

Distinguishing users from network nodes: In our model, the network (see Figure 1) is represented by a directed graph $G = (V, E)$, where the vertex set V is the union of two disjoint subsets, U and N. U is the set of *network users*, that is, originators and receivers of traffic, and N is the the set of *network nodes*, that is, the routers in the network. The term "users" here refers to users of the network and not to end-users. In an ISP network, the set of users U represents routers in neighboring networks ("adjacent routers") and the set of network nodes N represents the routers in the ISP's internal network ("internal routers").

User edges are potential links: To capture the ability to migrate, we introduce the notion of *potential links* that represent the locations where the user can *possibly* connect to the network. The edge set E is the union of two disjoint subsets, E_U and E_N, where $E_U \subseteq U \times N$ is the subset of edges connecting users to network nodes, and $E_N \subseteq N \times N$ is the subset of edges connecting network nodes to other network nodes. Each edge $e \in E_N$ has capacity $c_e \geq 0$, which measures the amount of flow that can traverse edge

Notation	Description
G	Network graph $G = (V, E)$
V	Network vertex, union of U and N
E	Network edge, union of E_U and E_N
U	Set of network users
N	Set of network nodes
E_U	Subset of edges that connect user $u \in U$ to network nodes in N, $E_U \subseteq U \times N$
E_N	Subset of edges that connect network node $n \in N$ to network nodes in N, $E_N \subseteq N \times N$
c_e	capacity of edge $e \in E$
L_u	Potential links, $L_u \subseteq E_U$
D	Demand matrix, $D = \{d_{ij}\}_{i,j \in U}$
d_{ij}	Amount of traffic that user i wishes to send user j
ϕ_e	cost function used in TLU minimization, function of f_e/c_e
f_e	Total flow traversing the edge e

Table 1: Summary of notation used in model of traffic engineering with migration.

e. We impose no capacity constraints on the edges in E_U (*i.e.,* edges have infinite capacity). We call the set of all edges $L_u \subseteq E_U$ that connect user $u \in U$ to network nodes in N "the set of u's *potential links*" (that is, $\forall u \in U$, $L_u = \{e = (u, v) |\ e \in E_U\}$).

In ISP networks, the set of potential links L_u for each adjacent router (user) u represents the points at which u can connect to the ISP network. This can, in practice, depend on the underlying transport network that can, for example, limit a user to connecting only to network nodes in nearby geographical regions. In addition, the set of potential links can reflect latency considerations, *e.g.*, it is beneficial to home frequently-communicating users near each other.

Demand matrix is user-to-user: Our model distinguishes network users from network nodes, and our demand matrix captures this distinction; we are now given a demand matrix $D = \{d_{ij}\}_{i,j \in U}$, where each entry d_{ij} specifies the amount of traffic *user* i wishes to send *user* j.

Each user must use a single potential link: The high-level goal is, for every pair of users i and j such that $d_{ij} > 0$, to distribute flow from i to j between the routes from i to j in G, subject to the constraint that every user can only connect to the network via a *single* link. That is, for every user $u \in U$, traffic flowing from that user to the other users, and vice versa, can only traverse a single edge in L_u; traffic along all other edges in L_u must be 0. When optimizing the flow of traffic through the network we again consider the TLU minimization objective function.

2.3 Practical Considerations

Naturally, our formal framework does not capture all real-life constraints, e.g., the cost of migration (in terms of processing, offline time, and more), physical limitations of the individual vertices in the network (including the number and capacity of links that each vertex can support), and the cost to use different potential links. Each of these considerations can be incorporated into our model through introducing additional variables and constraints and is left as future work.

We point out that while we do not explicitly model multi-homing, the common practice that a user connects to the network at multiple locations, our framework and results are also applicable to this scenario. With router grafting, an ISP does not control how a user splits traffic over multiple connections – a user with multiple connections can be modeled as multiple users with a single connection.

Finally, we note that while our model does not impose restrictions on the capacity of potential links, our simulations use real traffic patterns which induce bounds on how much traffic can be sent and received along each potential link.

3. EDGE-LINK MIGRATION HEURISTICS

Ideally, we would be able to find a TLU-minimizing solution in which each user sends/receives traffic along a single potential link in a computationally-efficient manner. Unfortunately, we prove that finding an optimal solution is NP-hard and that even approximating the optimal solution within a constant factor is intractable. We thus seek heuristics which fare well in practice. We present two heuristics—the *max-link heuristic* and the *cluster-user heuristic*. We experimentally evaluate these heuristics in Section 4.

3.1 "Good" Solutions are Hard to Compute

Computing an optimal solution is hard. In the traditional traffic-engineering setting (see Section 2.1), where migration is not considered, existing algorithms for computing multicommodity flow provide the optimal solution in a computationally-efficient manner. Unfortunately, computing the multicommodity flow in our "TE with migration" setting is computationally intractable. In fact, this is true even when every user has at most two potential links. We show a full proof based on a reduction from the NP-hard PARTITION problem in an extended version of this paper [9].

THEOREM 3.1. *Minimizing TLU subject to the restriction that each user send/receive traffic along a single potential link is NP-hard even when $|L_u| \leq 2$ for every user u.*

Approximating the optimal solution is also hard In the extended version of this paper [9] we also prove, based on a reduction from MAX-LABEL-COVER, that even approximating the optimum within a constant factor better than 1000 is computationally intractable.

THEOREM 3.2. *Approximating the TLU minimizing solution within a factor better than* 1000 *subject to the restriction that each user send/receive traffic along a single potential link is NP-hard.*

3.2 Max-Link Heuristic

Our first heuristic is the max-link heuristic. Intuitively, the max-link heuristic first computes the maximum multicommodity flow in the input network that contains *all* potential links. Then, the heuristic uses this fully-fractional flow (where users' traffic can be split between all their potential links) to choose a single potential link for each user, thus constructing a feasible (integral) solution. To do this, the max-link heuristic throws away, for each user, all potential links but the single potential link along which the user sends and receives the most traffic.

3.3 Cluster-User Heuristic

We now present the cluster-user heuristic, which considers the users in groups, and not individually. With the cluster-user heuristic, the set of users is first divided into groups (or clusters) of users, such that every cluster contains a number of users who all can connect to the network at the exact same locations (network nodes). We then create a new network where each cluster of users is replaced by a single user (with a set of potential links that connect to the network at the exact same network nodes). We then solve multicommodity flow for this network (allowing traffic to be split between multiple potential links) to get the fraction of traffic flowing over each potential link. We then use the multicommodity flow solution to map users to potential links with the goal of matching the split of traffic over the potential links as closely as possible. Observe that this is only a "best fit", as splitting the traffic exactly as in the multicommodity flow may not be possible. We note that, in general, finding the best fit can easily be shown to be NP-hard, even in the case that every user has 2 potential links, yet good approximations are achievable. In our experiments, we were able to use a brute-force approach to determine the best fit.

The intuition behind the cluster-user heuristic lies in the idea that networks will have a large number of users who all can connect to the network at the exact same locations, and that each user's demands constitute a small fraction of the demands of the cluster as a whole. We observe that, in this case, each cluster can be regarded as a single user with the ability to split traffic among its potential links almost as in the optimal multicommodity flow solution. This follows from the fact each user in the cluster sends/receives a negligible amount of the total traffic, and so users in the cluster can be mapped to outgoing links so as to closely mimic the multicommodity flow solution.

4. EVALUATING TE WITH MIGRATION

The goal of this evaluation is to demonstrate the benefits of using migration in traffic engineering, even with simple heuristics. We first show in Section 4.2 that our max-link heuristic does indeed lead to an improvement in network performance. We then examine two additional concerns relating to more practical questions—how often do links need to be migrated (Section 4.3) and how many links need to be migrated (Section 4.4).

4.1 Experimental Setup with Internet2 Data

We based all of our experiments on data collected from Internet2 [6], which consists of $N = 9$ core routers and $U = 205$ external routers. We collected one week of data starting January 18, 2010. From each router, we downloaded the previously collected NetFlow data which provides summaries of the sampled flows (at the rate of 1/100 packets) in 5-minute intervals (1-week of traffic is 2016 5-minute samples). We also downloaded the routing information base (RIB) and the output for the '*show bgp neighbor*' command, both of which are captured every two hours. Every NetFlow entry contains the incoming interface, which we used to represent an external source user. We used the routing tables for each of the routers to determine the egress router for each flow, along with the specific interface on the egress router that the flow exits the network on, which we used to represent the external destination user. This enabled us to generate an external-user-to-external-user traffic matrix.

Our choice of the set of potential links (the L_u's) was based on geographical distance. The first potential link for a given user is to the router the user is connected to in the original topology. The second potential link is to a router randomly selected from the routers nearest the original [1].

4.2 Migration Improves Network Utilization

The first metric of importance is simply the improvement that can be obtained when utilizing link migration. Here we first define the metric we are evaluating, show that the improvement varies depending on the traffic patterns, and conclude that max-link is slightly better.

4.2.1 *Defining 'Improvement'*

In order to determine how much improvement we can achieve over a network optimizing only the routing we need to clearly define a metric for which we will compare. For this, consider the total link utilization for an example 5-minute period shown in Figure 2. The Figure shows results for the original (optimally engineered) network (the "original topology" line), and for traffic engineering with migration (using max-link) with 2 links per user (the "optimized topology" line).

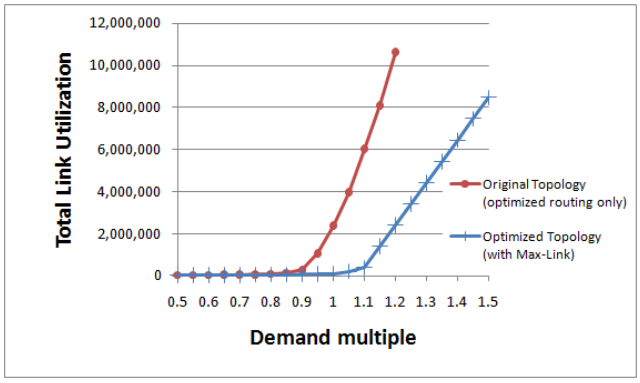

Figure 2: Evaluation of max-link for a single 5-minute period.

To obtain the graph in Figure 2, we varied the traffic demand by scaling all entries in the demand matrix by a multiplicative factor, plotted on the x-axis, and optimized for the TLU for each. TLU minimization captures the goal of avoiding congestion, and involves an exponentially increasing cost for utilizing a link (see Section 2). We used the cost function from [5] as detailed in Section 2.1.

Due to the exponentially increasing cost, the network operator will wish to be at a point in the curve that comes before the exponential rise, that is, before the "knee" in the curve. Observe that this "knee" shifted to the right by roughly 20%, and so, with migration, the network can handle 20% more traffic with the same level of congestion.

Note that the original topology is currently operating at the knee (since the knee is at 1, which is the actual demand matrix). From this, we define the improvement met-

[1]We do not present results for more than two potential links per user. If two users have a potential link that connects to the same router, all traffic between those users will simply go through that router. As the number of potential links per user approaches the total number of core routers, each pair of users will have a router in common and a multicommodity flow solution will give no guidance on which links to use.

ric as the amount of traffic the network can carry in the optimized topology at the same level of congestion as the original topology—where the TLU represents the level of congestion. So, from the original topology, we found the minimal TLU with a demand multiple of 1 (i.e., the actual amount of traffic). We then determined which demand multiple in the optimized topology (i.e., with migration) would result in the same TLU. In other words, in terms of the graph in Figure 2, we found the y value for x=1 on the original topology line, and used that y value to find the x value on the optimized topology line.

Not all periods will see an equal improvement. The traffic patterns dictate how much improvement can be achieved—if the network is not that congested during a particular interval, performing edge link migration will have minimal affect since, in that case, it effectively is optimizing an already underutilized network.

4.2.2 Max-link is Slightly Better Than Cluster-user

Shown in Table 2 is a comparison of the improvements achieved with the max-link heuristic as well as the cluster-user heuristic, for different cluster sizes. From this we can see on average, traffic engineering with migration can increase network utilization by about 18.8% when using max-link, and slightly less when using cluster-user. Note that in both the max-link and cluster-user heuristics, we first calculate the TLU minimizing multicommodity flow of a graph which includes the potential links. The input is a prediction of what the expected demands will be so that a new topology can be optimized for it. For the experiments, we utilized the actual demand matrix, in essence giving perfect predictive power. We rely on an ISP's ability to predict traffic based on past history and the improvement obtainable will be related to the accuracy of the prediction (as we show in Section 4.3, we can still achieve good results over longer intervals, which are easier to predict).

Intuitively this improvement comes from two factors. The first is that by optimizing the homing location based on the demand matrix, users that communicate will tend to get closer together. Without link migration, the homing location must be determined up front and then cannot change. With link migration, we can alter the topology to bring users that communicate a lot closer together. The second factor is that by re-optimizing the topology, we can have a significant impact on congested links. By giving some traffic the ability to avoid the congested link (through migration), we can reduce the congestion on that link. There are, however, a small number of cases (2.6% in the case of max-link) where migrating links actually decreased performance. As mentioned in Section 3 these are only heuristics which fare well in practice but do not approximate the optimal result within some factor. Therefore, it is expected that there can be conditions which result in poor performance.

The cluster-user heuristic with a cluster size of two is very comparable to max-link. Within the cluster-user heuristic, the performance decreased as the cluster size increased. This suggest there are many factors related to the cluster size and cluster contents which influence performance as the fitting penalty decreases with the increase in cluster sizes—a lower best-fit penalty is better as it means we were able to match the optimal multi-commodity flow results more closely. Further research is needed to determine the best cluster composition.

heuristic	min	max	mean	#intervals worse (frac.)
max-link	0.783	1.550	1.188	54 (0.0268)
cluster-2	0.860	1.550	1.172	22 (0.0109)
cluster-4	0.565	1.550	1.121	76 (0.0377)
cluster-6	0.718	1.480	1.080	208 (0.1032)
cluster-8	0.790	1.460	1.066	261 (0.1295)
cluster-10	0.622	1.363	1.051	375 (0.1860)

Table 2: Comparison of the improvement over the original topology optimized for routing only when with max-link and cluster-user (for different cluster sizes over 7 days of traffic with 5-minute intervals).

4.3 Frequent Migration is Not Necessary

In Section 4.2, we examined the benefits of utilizing link migration in traffic engineering. We looked at the benefits when we could migrate every interval and knew the traffic in the next interval. However, predicting this can be difficult on that short of a time scale. Here, we examine how frequently we really need to be migrating.

To determine how often migration should occur, we looked at different periods—every 5 minutes, 30 minutes, 1 hour, 6 hours, 12 hours, and 24 hours. The demand matrix used when computing the multicommodity flow in the max-link heuristic (i.e., the predicted traffic), was the average demand matrix for the next interval (e.g., the next 6 hours). This was used to determine the optimized topology that would be used for the entire interval. As with the 5-minute case, errors in the prediction affect the results. We note, however, that as the intervals become longer, traffic patterns smooth out and become more predictable. We determined the TLU for each 5-minutes of traffic using this topology and compared the results to the original topology.

Table 3 shows the results for the different intervals. As could be expected, the longer the interval, the worse the results. However, even re-optimizing the topology every 6 or 12 hours still has good performance.

interval	min	max	mean	#worse (frac.)
5 mins	0.783	1.550	1.188	54 (0.0267)
30 mins	0.757	1.550	1.166	146 (0.0724)
1 hour	0.777	1.550	1.163	152 (0.0753)
6 hours	0.801	1.550	1.149	182 (0.0902)
12 hours	0.856	1.550	1.141	191 (0.0947)
24 hours	0.806	1.550	1.083	465 (0.2306)

Table 3: Comparison of the improvement over the original topology optimized for routing only when performing grafting at different intervals (over 7 days traffic).

4.4 Only a Few Links Need to be Migrated

Our formulation of traffic engineering with migration does not currently incorporate the cost of migration. To decide which users to migrate, we can weigh the cost of migrating a user against the gain from migrating that user; when the impact of migrating a user is low (*e.g.*, when that user generates and consumes negligible amounts of traffic), migration might be undesirable. To investigate this, we examined the amount of traffic each user sends or receives for an example 5-minute interval. We found that 85% of the traffic comes

from just 42 (out of 205) users, of which, max-link only determined 5 of them should be migrated. Hence, we can still obtain a significant improvement in network performance while migrating only a small number of links.

To determine the effect across the entire data sample, we examined three different thresholds—100% (i.e., migrate all links that max-link determined need to be migrated), 95%, and 90%. We found that on average the number of links to be migrated are 22.6 (for 100% threshold), 11.9 (for 95% threshold), and 9.2 (for 90% threshold). Therefore, by not worrying about a small fraction of traffic, we can greatly reduce the number of links that need to be migrated.

5. RELATED WORK

Due to its importance for network performance, traffic engineering has been heavily studied. There has been much work on schemes for traffic engineering in ISP networks [16] [7] [4] [3] [17]. This work interprets traffic engineering as the adaptation of the routing of traffic within the network so as to optimize performance. We, in contrast, *also* explore how to adapt traffic's ingress and egress points.

User actions can also change the traffic matrix – *e.g.,* using overlay routing to circumvent congested links [2, 13] changes the offered load. However, such "selfish" overlay routing can greatly *reduce* the effectiveness of traffic engineering [11].

Previous work on interdomain traffic engineering [10, 12, 14, 15] considers how to select among a group of fixed egress points for directing traffic to neighboring domains. Interdomain traffic engineering changes the traffic and BGP routes seen in neighboring networks, whereas edge-link migration is transparent. Also, the optimization approaches differ. Interdomain traffic engineering splits traffic over multiple edge links, whereas our "TE with migration" approach selects a *single* edge link for all traffic to and from each user.

Our work builds on earlier research proposing mechanisms for re-homing customers [1] [8]. However, these papers did not explore the implications for traffic engineering.

6. CONCLUSIONS AND FUTURE WORK

We proposed a new approach to traffic engineering where instead of only optimizing for fixed (predicted) traffic patterns, we also influence where traffic enters and exits the network. We showed that while computing an optimal solution is hard, even relatively simple heuristics can lead to significant performance gains without requiring frequent migration or large numbers of links to migrate. We view our work as a first step in this direction. Incorporating more practical aspects of traffic engineering into our model is a promising direction for future research. Further exploring cluster selection in the cluster user heuristic is also left for future work. Finally, while we focused on edge-link migration in ISP networks, we note that similar capabilities exist in data center networks—namely, virtual machine migration. Exploring our framework in data-center networks, which have different patterns and practical constraints, is also an area for future research.

7. REFERENCES

[1] M. Agrawal, S. Bailey, A. Greenberg, J. Pastor, P. Sebos, S. Seshan, J. van der Merwe, and J. Yates. RouterFarm: Towards a dynamic, manageable network edge. In *SIGCOMM Workshop on Internet Network Management*, September 2006.

[2] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris. Resilient overlay networks. In *Proc. ACM SOSP*, October 2001.

[3] D. Applegate, L. Breslau, and E. Cohen. Coping with network failures: Routing strategies for optimal demand oblivious restoration. In *Proc. ACM SIGCMETRICS*, June 2004.

[4] A. Elwalid, C. Jin, S. Low, and I. Widjaja. MATE: MPLS adaptive traffic engineering. In *Proc. IEEE INFOCOM*, 2001.

[5] B. Fortz and M. Thorup. Internet traffic engineering by optimizing OSPF weights. In *Proc. IEEE INFOCOM*, 2000.

[6] Internet2. `http://www.internet2.org`.

[7] S. Kandula, D. Katabi, B. Davie, and A. Charny. Walking the tightrope: Responsive yet stable traffic engineering. In *Proc. SIGCOMM*, 2005.

[8] E. Keller, J. Rexford, and J. van der Merwe. Seamless BGP session migration with router grafting. In *Proc. Networked Systems Design and Implementation*, April 2010.

[9] E. Keller, M. Schapira, and J. Rexford. Rehoming Edge Links for Better Traffic Engineering. Technical Report TR-917-11, Princeton University Computer Science Department, 2011.

[10] R. Mahajan, D. Wetherall, and T. Anderson. Negotiation-based routing between neighboring ISPs. In *Proceedings of the 2nd Symposium on Networked Systems Design and Implementation*, Boston, MA, USA, April 2005.

[11] L. Qiu, Y. R. Yang, Y. Zhang, and S. Shenker. Selfish routing in Internet-like environments. In *Proc. SIGCOMM*, 2003.

[12] M. Roughan and Y. Zhang. GATEway: symbiotic inter-domain traffic engineering'. In *The Second International Workshop on Game Theory in Communication Networks*, Athens, Greece, October 2008.

[13] S. Savage, T. Anderson, A. Aggarwal, D. Becker, N. Cardwell, A. Collins, E. Hoffman, J. Snell, A. Vahdat, G. Voelker, and J. Zahorjan. Detour: A case for informed Internet routing and transport. *IEEE Micro*, January 1999.

[14] R. Szabó, A. Takács, and A. Császár. Optimised multi homing – an approach for inter-domain traffic engineering. In *Proceedings of the 2nd International Workshop on Inter-Domain Performance and Simulation (IPS2004)*, Budapest, Hungary, March 2004.

[15] R. Teixeira, T. Griffin, M. G. C. Resende, and J. Rexford. TIE breaking: Tunable interdomain egress selection. *IEEE/ACM Trans. Networking*, August 2007.

[16] H. Wang, H. Xie, L. Qiu, Y. R. Yang, Y. Zhang, and A. Greenberg. COPE: Traffic engineering in dynamic networks. In *Proc. SIGCOMM*, 2006.

[17] C. Zhang, Z. Ge, J. Kurose, Y. Liu, and D. Towsley. Optimal routing with multiple traffic matrices: Tradeoff between average case and worst case performance. In *Proc. International Conference on Network Protocols*, Nov. 2005.