Optimizing Cost and Performance in Online Service Provider Networks

Zheng Zhang
Purdue University

Ming Zhang
Microsoft Research

Albert Greenberg
Microsoft Research

Y. Charlie Hu
Purdue University

Ratul Mahajan
Microsoft Research

Blaine Christian
Microsoft Corporation

Abstract– We present a method to jointly optimize the cost and the performance of delivering traffic from an online service provider (OSP) network to its users. Our method, called Entact, is based on two key techniques. First, it uses a novel route-injection mechanism to measure the performance of alternative paths that are not being currently used, without disturbing current traffic. Second, based on the cost, performance, traffic, and link capacity information, it computes the optimal cost *vs.* performance curve for the OSP. Each point on the curve represents a potential operating point for the OSP such that no other operating point offers a simultaneous improvement in cost and performance. The OSP can then pick the operating point that represents the desired trade-off (e.g., the "sweet spot"). We evaluate the benefit and overhead of Entact using trace-driven evaluation in a large OSP with 11 geographically distributed data centers. We find that by using Entact this OSP can reduce its traffic cost by 40% without any increase in path latency and with acceptably low overheads.

1 Introduction

Providers of online services such as search, maps, and instant messaging are experiencing an enormous growth in demand. Google attracts over 5 billion search queries per month [2], and Microsoft's Live Messenger attracts over 330 million active users each month [5]. To satisfy this global demand, online service providers (OSPs) operate a network of geographically dispersed data centers and connect with many Internet service providers (ISPs). Different users interact with different data centers, and ISPs help the OSPs carry traffic to and from the users.

Two key considerations for OSPs are the cost and the performance of delivering traffic to its users. Large OSPs such as Google, Microsoft, and Yahoo! send and receive traffic that exceeds a petabyte per day. Accordingly, they bear huge costs to transport data.

While cost is clearly of concern, performance of traffic is critical as well because revenue relies directly on it. Even small increments in user-experienced delay (e.g., page load time) can lead to significant loss in revenue through a reduction in purchases, search queries, or advertisement click-through rates [20]. Because application protocols involve multiple round trips, small increments in path latency can lead to large increments in user-experienced delay.

The richness of OSP networks makes it difficult to optimize the cost and performance of traffic. There are numerous destination prefixes and numerous choices for mapping users to data centers and for selecting ISPs. Each choice has different different cost and performance characteristics. For instance, while some ISPs are free, some are exorbitantly expensive. Making matters worse, cost and performance must be optimized jointly because the trade-off between the two factors can be complex. We show that optimizing for cost alone leads to severe performance degradation and optimizing for performance alone leads to significant cost.

To our knowledge, no automatic traffic engineering (TE) methods exist today for OSP networks. TE for OSPs requires a different formulation than that for transit ISPs or multihomed stub networks. In the traditional intra-domain TE for transit ISPs, the goal is to balance load across multiple internal paths [13, 18, 23]. End-to-end user performance is not considered.

Unlike multihomed stub networks, OSPs can source traffic from any of their multiple data centers. This flexibility adds a completely new dimension to the optimization. Further, large OSPs connect to hundreds of ISPs – two orders of magnitude more than multihomed stub networks – which calls for highly scalable solutions. Another assumption in TE schemes for multihomed sites [7, 8, 15] is that each connected ISP offers paths to all Internet destinations. This assumption is not valid in the OSP context.

Given the limitations of the current TE methods, the state of the art for optimizing traffic in OSP networks is rather rudimentary. Operators manually configure a delicate balance between cost and performance. Because of the complexity of large OSP networks, the operating point thus achieved can be far from desirable.

We present the design and evaluation of Entact, the first TE scheme for OSP networks. We identify and address two primary challenges in realizing such a scheme. First, because the interdomain routing protocol (BGP) does not include performance information, performance is unknown for paths that can be used but are not being currently used. We must estimate the performance of such paths without actually redirecting traffic to them as redirection can be disruptive. We overcome this challenge via a novel *route injection* technique. To measure an unused path for a prefix, Entact selects an IP address ip within the prefix and installs a route for $ip/32$ to routers in the OSP network. Because of the longest-prefix match rule, packets destined to ip will follow the installed route while the rest of the traffic will continue to use the current route.

The second challenge is to use the cost, performance, traffic volume, and link capacity information to find in real time a TE strategy that matches the OSP's goals. Previous algorithmic studies of route selection optimize one of the two metrics, performance or cost, with the other as the fixed constraint. However, from conversations with the operators of a large OSP, we learned that often there is no obvious answer for which metric should be selected as the fixed constraint, as profit depends on the complex trade-off between performance and cost. Entact uses a novel joint optimization technique that finds the entire trade-off curve and lets the operator pick a desirable point on that curve. Such a technique provides operators with useful insight and a range of options for configuring the network as desired.

We demonstrate the benefits of Entact in Microsoft's global network (MSN), one of the largest OSPs today. Because we are not allowed to arbitrarily change the paths used by various prefixes, we conduct a trace-driven study. We implement the key components of Entact and measure the relevant routing, traffic, and performance information. We use this information to simulate Entact-based TE in MSN. We find that compared to the common (manual) practices today, Entact can reduce the total traffic cost by up to 40% without compromising performance. We also find that these benefits can be realized with low overhead. Exploring two closest data centers for each destination prefix and one non-default route at each data center tends to be enough, and changing routes once per hour tends to be enough.

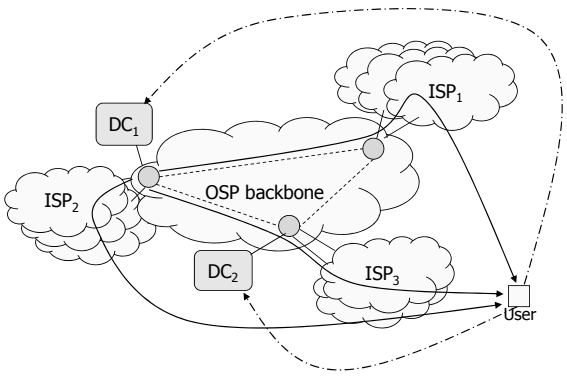

Figure 1: Typical network architecture of a large OSP.

2 Traffic Cost and Performance for OSPs

In this section, we describe the architecture of a typical OSP network. We also outline the unique cost and performance optimization opportunities that arise in OSP networks by exploiting the presence of a diverse set of alternative paths for transporting service traffic.

2.1 OSP network architecture

Figure 1 illustrates the typical network architecture of large OSPs. To satisfy global user demand, such OSPs have data centers (DCs) in multiple geographical locations. Each DC hosts a large number of servers, anywhere from several hundreds to hundreds of thousands. For cost, performance, and robustness, each DC is connected to many ISPs that are responsible for carrying traffic between the OSP and its millions of users. Large OSPs such as Google and Microsoft often also have their own backbone network to interconnect the DCs.

2.2 Cost of carrying traffic

The traffic of an OSP traverses both internal links that connect the DCs and external links that connect to neighboring ISPs. The cost model is different for the two types of links. The internal links are either dedicated or leased. Their cost is incurred during acquisition, and any recurring cost is independent of the traffic volume that they carry. Hence, we can ignore this cost when engineering an OSP's traffic.

The cost of an external link is a function of traffic volume, i.e., $F(v)$, where F is a non-decreasing cost function and v is the charging volume of the traffic. The cost function F is commonly of the form $price \times v$, where $price$ is the unit traffic volume price of a link. The charging volume v is based on actual traffic volume. A common practice is to use the 95th-percentile (*P95*). Under

this scheme, the traffic volume on the link is sampled for every 5-minute interval. At the end of a billing period, *e.g.*, a month, the charging volume is the 95th percentile across all the samples. Thus, the largest 5% of the intervals are not considered, which protects an OSP from being charged for short bursts of traffic.

In principle, the charging volume is the maximum of the P95 traffic in either direction. However, since user requests tend to be much smaller than server replies for online services, the outgoing direction dominates. Hence, we ignore inbound traffic when optimizing the cost of OSP traffic.

2.3 Performance measure of interest

There are several ways to measure the user-perceived performance of an online service. In consultation with OSP operators, we use round trip time (RTT) as the performance measure, which includes the latency between the DC and the end host along both directions. The performance of many online services, such as search, email, maps, and instant messaging, is latency-bound. Small increments in latency can lead to significant losses in revenue [20] .

Some online services may also be interested in other performance measures such as available bandwidth or loss rate along the path. A challenge with using these measures for optimizing OSP traffic is scalable estimation of performance for tens of thousands of paths. Accurate estimation of available bandwidth or loss rate using current techniques requires a large number of probes [17, 19, 25]. We leave for the future the task of extending our work to other performance measures.

2.4 Cost-performance optimization

A consequence of the distributed and rich connectivity of an OSP network is that an OSP can easily have more than a hundred ways to reach a given user in a destination prefix. First, an OSP usually replicates an online service across multiple DCs in order to improve user experience and robustness. An incoming user request can thus be directed to any one of these DCs, *e.g.*, using DNS redirection. Second, the traffic to a given destination prefix can be routed to the user via one of many routes, either provided by one of the ISPs that directly connect to that DC or by one of the ISPs that connect to another DC at another location (by first traversing internal links). Assuming P DCs and an total of Q ISPs, the number of possible *alternative paths* for a request-response round trip is $P * Q$. (An OSP can select which DC will serve a destination prefix, but it typically does not control which link is used by the incoming traffic.)

The large number of possible alternative paths and differences in their cost and performance creates an opportunity for optimizing OSP traffic. This optimization needs to select the target DC and the outgoing route for each destination prefix. The (publicly known) state-of-the-art in optimizing OSP traffic is mostly manual and ad hoc. The default practice is to map a destination prefix to a geographically close DC and to let BGP control the outgoing route from that DC. BGP's route selection is performance-agnostic and can take cost into account in a coarse manner at best. On top of that, exceptions may be configured manually for prefixes that have very poor performance or very high cost.

The complexity of the problem, however, limits the effectiveness of manual methods. Effective optimization requires decisions based on the cost-performance trade-offs of hundreds of thousands of prefixes. Worse, the decisions for various prefixes cannot be made independently because path capacity constraints create complex dependencies among prefixes. Automatic methods are thus needed to manage this complexity. The development of such methods is the focus of our work.

3 Problem Formulation

Consider an OSP as a set of data centers $DC = \{dc_i\}$ and a set of external links $LINK = \{link_j\}$. The DCs may or may not be interconnected with backbone links. The OSP needs to deliver traffic to a set of destination prefixes $D = \{d_k\}$ on the Internet. For each d_k, the OSP has a variety of paths to route the request and reply traffic, as illustrated in Figure 2. A *TE strategy* is defined as a collection of assignments of the traffic (request and reply) for each d_k to a $path(dc_i, link_j)$. Each assignment conceptually consists of two selections, namely *DC selection*, *e.g.*, selecting a dc_i, and *route selection*, *e.g.*, selecting a $link_j$. The assignments are subject to two constraints. First, the traffic carried by an external link should not exceed its capacity. Second, a prefix d_k can use $link_j$ only if the corresponding ISP (which may be a peer ISP instead of a provider) provides routes to d_k.

Each possible TE strategy has a certain level of aggregate performance and incurs certain traffic cost to the OSP. Our goal is to discover the optimal TE strategies that represent the cost-performance trade-offs desired by the OSP. For instance, the OSP might want to maximize performance for a given cost. Additionally, the relevant inputs to this optimization are highly dynamic. Path performance as well as traffic volume of a prefix, which determines cost, change with time. We thus want an efficient, online scheme that adapts the TE strategy as the inputs evolve.

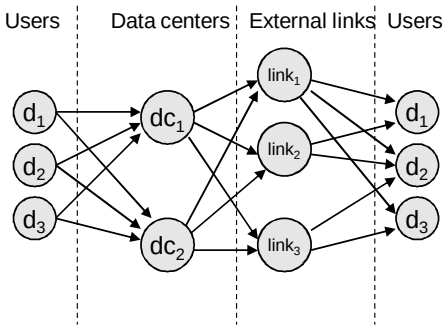

Figure 2: OSP traffic engineering problem.

4 Entact Key Techniques

In this section, we provide an overview of the key techniques in Entact. We present the details of their implementations in the next section. There are two primary challenges in the design of an online TE scheme in a large OSP network. The first challenge is to measure in real time the performance and cost of routing traffic to a destination prefix via any one of its many *alternative paths* that are not currently being used, without actually redirecting the current traffic to those alternative paths. Further, to keep up with temporal changes in network conditions, this measurement must be conducted at sufficiently fine granularity. The second challenge is to use that cost-performance information in finding a TE strategy that matches the OSP's goals.

4.1 Computing cost and performance

To quantify the cost and performance of a TE strategy, we first measure the performance of individual prefixes along various alternative paths. This information is then used to compute the aggregate performance and cost across all prefixes.

4.1.1 Measuring performance of individual prefixes

Our goal is to measure the latency of an alternative path for a prefix with minimal impact on the current traffic, *e.g.,* without actually changing the path being currently used for that prefix. One possible approach is to infer this latency based on indirect measurements. Previous studies have proposed various techniques for predicting the latency between two end points on the Internet [10,14,22,27]. However, they are designed to predict the latency of the current path between two end points in the Internet, and hence are not applicable to our task of measuring alternative paths.

We measure the RTT of alternative paths directly using a novel *route injection* technique. To measure an alternative path which uses a non-default route R for prefix p, we select an IP address ip within p and install the route R for $ip/32$ in the network. This special route is installed to the routers in the OSP by a BGP daemon that maintains iBGP peering sessions with them. Because of the longest-prefix match rule, packets destined to ip will follow the route R and the rest of the traffic will follow the default route. Once the alternative route is installed, we can measure the RTT to p along the route R using data-plane probes to ip (details in §5.1). Simultaneous measurements of multiple alternative paths can be achieved by choosing a distinct IP address for each alternative path.

4.1.2 Computing performance of a TE strategy

The measurements of individual prefixes can be used to compute the aggregate performance of any given TE strategy. We use the weighted average RTT ($wRTT$), $\frac{\sum vol_p \times RTT_p}{\sum vol_p}$, of all the traffic as the aggregate performance measure, where vol_p is the volume of traffic to prefix p, and RTT_p is the RTT of the path to p in the given TE strategy. The traffic volume vol_p is estimated based on the Netflow data collected in the OSP.

4.1.3 Computing cost of a TE strategy

A challenge in optimizing traffic cost is that the actual traffic cost is calculated based on the 95% link utilization over a long billing period (*e.g.,* a month), while an online TE scheme needs to operate at intervals of minutes or hours. While there exist online TE schemes that optimize P95 traffic cost [15], the complexity of such schemes makes them inapplicable to a large OSP network with hundreds of neighbor ISPs. We thus choose to only consider short-term cost in TE optimization rather than directly optimizing P95 cost. Our hypothesis is that, by consistently employing low-cost strategies in each short interval, we can lower the actual traffic cost over the billing period. We present results that validate this hypothesis in §7.

We use a simple computation to quantify the cost of a TE strategy in an interval. As discussed in §2.2, we need to focus only on the external links. For each external link L, we add the traffic volume to all prefixes that choose that link in the TE strategy, *e.g.,* $Vol_L = \sum_p vol_p$, where prefix p uses link L for vol_p amount of traffic. The total traffic cost of the OSP is $\sum_L F_L(Vol_L)$, where $F_L(.)$ is the pricing function of the link L. Because this measure of cost is not the actual traffic cost over the billing period, we refer to this measure as *pseudo cost*.

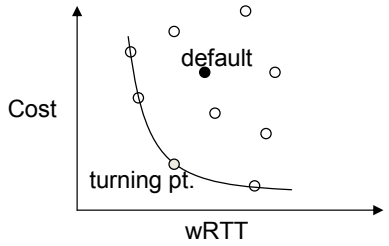

Figure 3: The cost-performance tradeoff in TE strategy space.

4.2 Computing optimal TE strategies

We now present our optimization framework that uses the cost and performance information to derive the desirable TE strategy for an OSP. We first assume the traffic to a destination prefix can be arbitrarily divided among multiple alternative paths and obtain a class of *optimal* TE strategies. In this class of strategies, one cannot improve performance without sacrificing cost or vice versa. Second, we describe how we select a strategy in this class that best matches the cost-performance trade-off that the OSP desires. Third, since in practice the traffic to a prefix cannot be arbitrarily split among multiple alternative paths, we devise an efficient heuristic to find an integral solution that approximates the desired fractional one.

4.2.1 Searching for optimal strategy curve

Given a TE strategy, we can plot its cost and performance (weighted average RTT or $wRTT$) on a 2-D plane. This is illustrated in Figure 3 where each dot represents a strategy. The number of strategies is combinatorial, ${N_p}^{N_a}$ for N_p prefixes and N_a alternative paths per prefix. A key observation is that not all strategies are worth exploring. In fact, we only need to consider a small subset of *optimal* strategies that form the lower-left boundary of all the dots on the plane. A strategy is optimal if no other strategy has both lower $wRTT$ and lower cost. Effectively, the curve connecting all the optimal strategies forms an *optimal strategy curve* on the plane.

To compute this curve, we sweep from a lower bound on possible $wRTT$ values to an upper bound on possible $wRTT$ values at small increments, *e.g.*, 1 ms, and compute the minimum cost for each $wRTT$value in this range. These bounds are set loosely, *e.g.*, the lower bound can be zero and the upper bound can be ten times the $wRTT$ of the default strategy.

Given a $wRTT$ R in this range, we compute the minimum cost using linear programing (LP). Following the notations in Figure 2, let f_{kij} be the fraction of traffic to d_k that traverses $path(dc_i, link_j)$ and rtt_{kij} be the RTT to d_k via $path(dc_i, link_j)$. The problem of computing cost can then be described as:

$$\text{min} \;\; pseudoCost = \sum_j (price_j \times \sum_k \sum_i (f_{kij} \times vol_k)),$$

subject to:

$$\sum_k \sum_i (f_{kij} \times vol_k) \leq \mu \times cap_j \quad (1)$$

$$\sum_k \sum_i \sum_j (f_{kij} \times vol_k \times rtt_{kij}) \leq \sum_k vol_k \times R \quad (2)$$

$$\sum_i \sum_j f_{kij} = 1 \quad (3)$$

$$\forall k, i, j \quad 0 \leq f_{kij} \leq 1 \quad (4)$$

Condition 1 represents the capacity constraint for each external link and μ is a constant (by default 0.95) that reserves some spare capacity to accommodate potential traffic variations for online TE. Condition 2 represents the $wRTT$ constraint. Condition 3 ensures all the traffic to a destination is carried. The objective is to find feasible values for variables f_{kij} that minimize the total pseudo cost. Solving such an LP for all possible values of R and connecting the TE strategy points thus obtained yield the optimal strategy curve.

4.2.2 Selecting a desirable optimal strategy

Each strategy on the optimal strategy curve represents a particular tradeoff between performance and cost. Based on its desired tradeoff, an OSP will typically be interested in one or more of these strategies. Some of these strategies are easy to identify, such as minimum cost for a given performance or minimum wRTT for a given cost budget. Sometimes, an OSP may desire a more complex tradeoff between cost and performance. For such an OSP, we take a parameter K as an input. This parameter represents the additional unit cost the OSP is willing to bear for a unit decrease in wRTT.

The desirable strategy for a given K corresponds to the point in the optimal strategy curve where the slope of the curve becomes higher than K when going from right to left. More intuitively, this point is also the "turning point" or the "sweet spot" when the optimal strategy curve is plotted after scaling the wRTT by K. We can automatically identify this point along the curve as the one with the minimum value of $pseudoCost + K \cdot wRTT$.

This point is guaranteed to be unique because the optimal strategy curve is convex. For convenience, we define $pseudoCost + K \cdot wRTT$ as the $utility$ of a strategy. Lower utility values are better. We can directly find this turning point by slightly modifying the original optimization problem to minimize utility instead of by solving the original optimization problem for all possible wRTT values.

4.2.3 Finding a practical strategy

The desirable strategy identified above assumes that traffic to a prefix can be split arbitrarily across multiple paths. In practice, however, the traffic to a prefix can only take one alternative path at a time, and hence variables f_{kij} must be either 0 or 1. Imposing this requirement makes the optimization problem an Integer Linear Programming (ILP) problem, which is NP-hard. We devise a heuristic to approximate the fractional solution to an optimal strategy with an integral solution. Intuitively, our heuristic searches for an integral solution "near" the desired fractional one.

We start with the fractional solution and sort all the destination prefixes d_k in the ascending order based on $avail_k = \sum_{j \in R_k} \lfloor \frac{availCap_j}{vol_k} \rfloor$, where vol_k is the traffic volume to d_k, R_k is the set of external links that have routes to reach d_k, and $availCap_j$ is the available capacity at $link_j$. The $availCap_j$ is initialized to be the capacity of $link_j$ and updated each time a prefix is assigned to use this link. The $avail_k$ measure gives high priority to prefixes with large traffic volume and small available capacity. We then greedily assign the prefixes to paths in the sorted order.

Given a destination d_k and its corresponding f_{kij}'s in the fractional solution, we randomly assign all of its traffic to one of the paths $path(dc_i, link_j)$ that has enough residual capacity for d_k with a probability proportional to f_{kij}. Compared to assigning the traffic to the path with the largest f_{kij}, random assignment is more robust to a bad decision for one particular destination. Once a prefix is assigned, the available capacity of the selected link is adjusted accordingly, and the $avail_k$-based ordering of the remaining unassigned prefixes is updated as well. In theory, better integral solutions can be obtained using more sophisticated methods [26]. But as we show later, our simple heuristic approximates the fractional solution closely.

5 Prototype Implementation

In this section, we describe our implementation of Entact. As shown in Figure 4, there are three inputs to Entact. The first input is Netflow data from all routers in the OSP network, which gives us information on flows currently traversing the network. The second input is routing tables from all routers, which gives us information not only on routes currently being used and but also on alternative routes offered by neighbor ISPs. The third input is the information on link capacities and prices. The output of Entact is a recommended TE strategy.

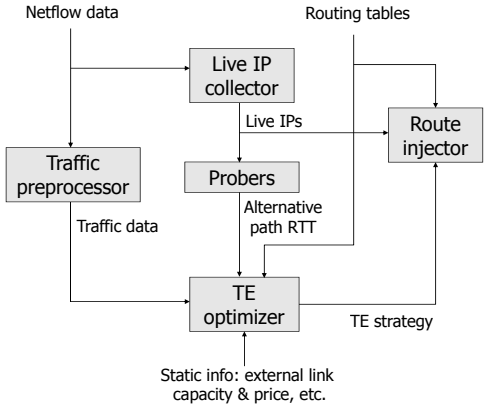

Figure 4: The Entact architecture

Entact divides time into fixed-length windows of size TE_{win} and a new output is produced in every window. To compute the TE strategy in window i, the measurements of traffic volume and path performance from the previous window are used. We assume that these quantities change at a rate that is much slower than TE_{win}. We later validate this assumption and also evaluate the impact of TE_{win}. The recommended TE strategy is applied to the OSP network by injecting the selected routes, similar to the route injection of /32 IP addresses.

5.1 Measuring path performance

As mentioned before, to obtain measurements on the performance of alternative paths to a prefix, we inject special routes to IP addresses in that prefix and then measure performance by sending probes to those IP addresses. We identify IP addresses within a prefix that respond to our probes using the *Live IP collector* component (Figure 4). The *Route Injector* component injects routes to those IP addresses, and the *Probers* measure the path performance. We describe each of these components below.

Live IP collector. Live IP collector is responsible for efficiently discovering IP addresses in a prefix that respond to our probes. A randomly chosen IP address in a prefix is unlikely to be responsive. We use a combination of two methods to discover live IP addresses. The first method is to probe a subset of IP addresses that are found in Netflow data. The second method is the heuristic proposed in [28]. This heuristic prioritizes and orders probes to a

small subset of IP addresses that are likely to respond, *e.g.*, *.1 or *.127 addresses, and hence is more efficient than random scanning of IP addresses.

Discovering one responsive IP address in a prefix is not enough; we need multiple IP addresses to probe multiple paths simultaneously and also to verify if the prefix is in a single geographical location (see §6.1). Even the combination of our two methods does not always find enough responsive IP addresses for every Internet prefix. In this paper, we restrict ourselves to those prefixes for which we can find enough responsive IP addresses. We show, however, that our results likely apply to all prefixes. In the future, we plan to overcome this responsive IP limitation by enlisting user machines, *e.g.*, through browser toolbars.

Route injector. Route injector selects alternative routes from the routing table obtained from routers in the OSP network, and installs the selected alternative routes on the routers. The route injector is a BGP daemon that maintains iBGP session with all core and edge routers in the OSP network. The daemon dynamically sends and withdraws crafted routes to those routers. We explain the details of the injection process using a simple example. We denote a path for a prefix p from data center DC as $path(DC, egress - nexthop)$, where $egress$ is the OSP's edge router along the path, and $nexthop$ is the ISP's next hop router that is willing to forward traffic from $egress$ to p. In Figure 5, suppose the default BGP route of p follows $path(DC, E_1 - N_1)$ and we have two other alternative paths. Given an IP address IP_2 within p, to measure an alternative path $path(DC, E_2 - N_2)$ we do the following,

- Inject $IP_2/32$ with nexthop as E_2 into all the core routers C_1, C_2, and C_3
- Inject $IP_2/32$ with nexthop as N_2 into E_2.

Now, traffic to IP_2 will traverse the alternative path that we want to measure, while all traffic to other IP addresses in p, *e.g.*, IP_1, will still follow the default path. Similarly, we can inject another IP address $IP_3/32$ within p and simultaneously measure the performance of the two alternative paths. With n IP addresses in a prefix, we can simultaneously measure the performance of n alternative paths from each DC. The route injection only needs to be performed once. The injected routes are re-used across all TE windows, and updated only when there are routing changes. If more than n paths need to be measured, we can divide a TE window into smaller slots, and measure only n paths in each slot. In this case, the route injector needs to refresh the injected routes for each slot.

We implement the daemon that achieves the above functionality by feeding configuration commands to drive `bgpd`, an existing BGP daemon [3]. We omit implementation details due to space limit. It is important, however, to note that the core and edge routers should be configured to keep the injected routes only to themselves. Therefore, route injection does not encounter route convergence problems, or trigger any route propagation in or outside the OSP network.

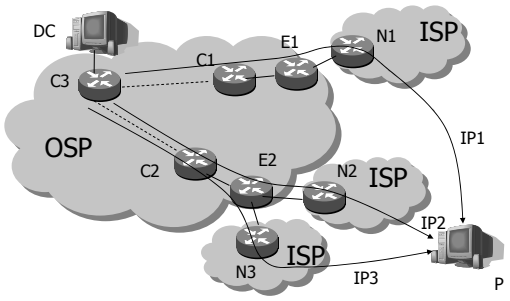

Figure 5: Route injection in a large OSP network.

Probers. Probers are located at all data centers in the OSP network and probe the live IPs along the selected alternative paths to measure their performance. For each path, a prober takes five RTT samples and uses the median as the representative estimate of that path. The probing module sends a TCP ACK packet to a random high port of the destination. This will often trigger the destination to return a TCP RST packet. Compared with using ICMP probes, the RTT measured by TCP ACK/RST is closer to the latency experienced by applications because ICMP packets may be forwarded in the network with lower priority [16].

5.2 Computing TE strategy

The computation of the TE strategy is based on the path performance data, the prefix traffic volume information, and the desired operating point of the OSP. The prefix traffic volume is computed by the *traffic preprocessor* component in Figure 4. It uses Netflow streams from all core routers and computes the traffic volume to each prefix by mapping each destination IP address to a prefix. For scalability, the Netflow data in our implementation is sampled at the rate of 1/1000.

Finally, the *TE optimizer* component implements the optimization process described in §4.2. It uses MOSEK [6] to solve the LP problems required to generate the optimal strategy. After identifying the optimal fractional strategy, the optimizer converts it to an integer strategy which becomes the output of the optimization process.

Figure 6: Location of the 11 DCs used in experiments.

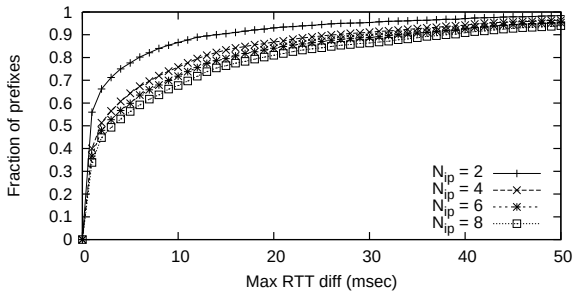

Figure 7: Maximum RTT difference among N_{ip} IPs within a prefix.

6 Experimental Setup

We conduct experiments in Microsoft's global network (MSN), one of the largest OSPs today. Figure 6 shows the location of the 11 MSN DCs that we use. These DCs span North America, Europe, and Asia Pacific and are inter-connected with high-speed dedicated and leased links that form the backbone of MSN. MSN has roughly 2K external links, many of which are free peering because that helps to lower transit cost for both MSN and its neighbors. The number of external links per DC varies from fewer than ten to several hundreds, depending on the location. We assume that services and corresponding user data are replicated to all DCs. In reality, some services may not be present at some of the the DCs. The remainder of this section describes how we select destination prefixes and how we quantify the performance and cost of a TE strategy.

6.1 Targeted destination prefixes

To reduce the overhead of TE, we focus on the high-volume prefixes that carry the bulk of traffic and whose optimization has significant effects on the aggregate cost and performance. We start with the top 30K prefixes which account for 90% of the total traffic volume. A large prefix advertised in global routing sometimes spans multiple geographical locations [21]. We could handle multi-location prefixes by splitting them into smaller sub-prefixes. However, as explained below, we would need enough live IP addresses in each sub-prefix to determine whether a sub-prefix is single-location or not. Due to the limited number of live IP addresses we can discover for each prefix (§5.1), we bypass the multi-location or low-volume prefixes in this paper.

We consider a prefix to be at a single location if the difference between the RTTs to any pair of IP addresses in it is under 5 ms. This is the typical RTT value between two nodes in the same metropolitan region [21]. A key parameter in this method is N_{ip}, the number of live IP addresses to which the RTTs are measured. On the one hand, we need to measure enough live IP addresses in order not to mis-classify a multi-location prefix as a single-location one. On the other hand, we can only identify a limited number of live IP addresses in a prefix.

Region	N.Amer.	Europe	A.Pac.	Lat.Amer.	Africa
%prefix	58	28	8	5	< 1
%traffic	59	29	6	6	< 1

Table 1: Locations of the 6K prefixes in our experiments.

To choose an appropriate N_{ip}, we examine the 4.1K prefixes that have at least 8 live IP addresses. Figure 7 illustrates the distributions of the maximum RTT difference of each of these prefixes as N_{ip} varies from 2 to 8. While the gap is significant between the distributions of N_{ip}=2 and N_{ip}=4, it becomes less pronounced as N_{ip} increases beyond 4. There is only an 8% difference between the distributions of N_{ip}= 4 and N_{ip}=8 when the maximum RTT difference is 5 ms. We thus pick $N_{ip} = 4$ to balance the accuracy of single-location prefix identification and the number of prefixes available for use.

After discarding prefixes with fewer than 4 live IP addresses, we are left with 15K prefixes. After further discarding prefixes that are deemed multi-location, we are left with 6K prefixes which we use in our study. Table 1 characterizes these prefixes by continents and traffic volumes. While a large portion of the prefixes and traffic are from North America and Europe, we also have some coverage in the remaining three continents. The prefixes are in 2,791 distinct ASes and account for 26% of the total MSN traffic. The number of alternative routes for a prefix varies at different DC locations. Among the 66K DC-prefix pairs, 61% have 1 to 4 routes, 27% has 5 to 8 routes, and the remaining 11% has more than 8 routes.

Our focus on a subset of prefixes raises two questions. First, are the results based on these prefixes applicable to

all prefixes? Second, how should we appropriately scale link capacities? We consider both questions next.

6.1.1 Representativeness of selected prefixes

We argue that the subset of prefixes that we study lets us estimate well the cost-performance trade-off for all traffic carried by MSN. For a given set of prefixes, the benefits of TE optimization hinge on the existence of alternative paths that are shorter or cheaper than the one used in the default TE strategy. We find that in this respect our chosen set of prefixes (P_s) is similar to other prefixes. We randomly select 14K high-volume prefixes (P_h) and 4K low-volume prefixes (P_l), which account for 29% and 0.8% of the total MSN traffic respectively. For each prefix p in P_h or P_l, we can identify 2 live IP addresses at the same location (with RTT difference under 5 ms). This means at least some sub-prefix of p will be at a single-location, even though p could span multiple locations.

For each prefix in P_s, P_h and P_l, we measure the RTT of the default route and three other randomly selected alternative routes from all the 11 DCs every 20 minutes for 1 day. We compare the default path used by the default TE strategy, *e.g.*, the path chosen by BGP from the closest DC, with all other 43 (may be fewer due to the availability of routes) alternative paths. Figure 8 illustrates the number of alternative paths that are better than the default path in terms of (a) performance, (b) cost, or (c) both. We see that the distributions are similar for the three sets of prefixes, which suggests that each set has similar cost-performance trade-off characteristics. Thus, our TE optimization results based on P_s are likely to hold for other traffic in MSN.

6.1.2 Scaling link capacity

Each external link has a fixed capacity that limits the traffic volume that it can carry. We extract link capacities from router configuration files in MSN. Because we only study a subset of prefixes, we must appropriately scale link capacities for our evaluation.

Let P_{all} and P_s denote the set of all the prefixes and the set of prefixes that we study. One simple approach is to scale down the capacity of all links by a constant $ratio = \frac{vol_{all}}{vol_s}$, where vol_{all} and vol_s are the traffic volumes of the two set of prefixes in a given period. The problem with this approach is that it overlooks the spatial and temporal variations of traffic, since $ratio$ actually depends on which link or which period we consider. This prompts us to compute a $ratio$ for each link separately. Our observation is that a link is provisioned for certain utilization level during peak time. Given $link_j$, we set $ratio_j = \frac{peak_j^{all}}{peak_j^s}$, where $peak_j^{all}$ and $peak_j^s$ are the peak traffic volume to P_{all} and to P_s under the default TE strategy during any 5-minute interval. This ensures the peak utilization of $link_j$ is the same before and after scaling. Note that $peak_{all}$ and $peak_s$ may occur in different 5-minute intervals.

Our method for scaling down link capacity is influenced by the default TE strategy. For instance, if $link_j$ never carries traffic to any prefix in P_s in the default strategy, its capacity will be scaled down to zero. This limits the alternative paths that can be explored in TE optimization, *e.g.*, any alternative strategies that use $link_j$ will not be considered even though they may help to lower wRTT and/or cost. Due to this limitation, our results, which show significant benefits for an OSP, actually represent a lower bound on the benefits achievable in practice.

6.2 Quantifying performance and cost

To quantify the cost of a given TE strategy, we record the traffic volume to each prefix and compute the traffic volume on each external link in each 5-minute interval. We then use this information to compute the 95% traffic cost (P95) over the entire evaluation period. Thus, even though Entact does not directly optimize for P95 cost, our evaluation measures the cost that the OSP will bear under the P95 scheme. We consider only the P95 scheme in our evaluation because it is the dominant charging model in MSN. Some ISPs do offer other charging models, such as long-term flat rate. Some ISPs also impose penalties if traffic volume falls below or exceeds a certain threshold. We leave for future work evaluating Entact under non-P95 schemes.

To quantify the performance, we compute the wRTT for each 5-minute interval and take the weighted average across the entire evaluation period. A minor complication is that we do not have fine time-scale RTT measurements for a prefix. To control overhead of active probing and route injection, we obtain two measurements (where each measurement is based on sending 5 RTT probes) in a 20-minute interval.

We find, however, that these coarse time-scale measurements are a good proxy for predicting finer time-scale performance. To illustrate this, we randomly select 500 prefixes and 2 alternate routes for each selected prefix. From each DC, we measure each of these 1,000 paths once a minute during a 20-minute interval. We then divide the interval into four 5-minute intervals. For each path and a 5-minute interval, we compute $\overline{rtt_5}$ by averaging the 5 measurements in that interval. For the same path, we also compute $\widetilde{rtt_{20}}$ by averaging two randomly selected measurements in the 20-minute interval. We conduct this experiment for 1 day and calculate the

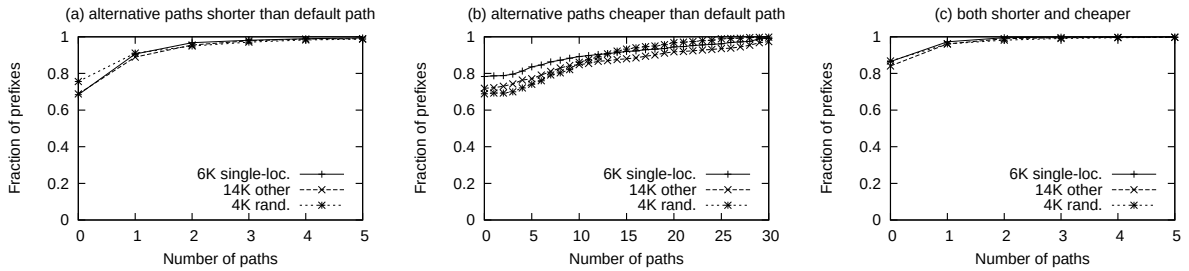

Figure 8: Number of alternative paths that are better than the default path in the set of 6K single-location prefixes, the set of 14K other high-volume prefixes, and the set of 4K randomly selected low-volume prefixes.

difference between $\overline{rtt}_5$ and $\widetilde{rtt}_{20}$ of all paths. It turns out that $\widetilde{rtt}_{20}$ are indeed very close to $\overline{rtt}_5$. The difference is under 1 ms and 5 ms in 78% and 92% of the cases respectively.

7 Results

In this section, we demonstrate and explain the benefits of online TE optimization in MSN. We also study how the TE optimization results are affected by a few key parameters in Entact, including the number of DCs, number of alternative routes, and TE optimization window. Our results are based on one-week of data collected in September 2009, which allows us to capture the time-of-day and day-of-week patterns. Since the traffic and performance characteristics in MSN are usually quite stable over several weeks, we expect our results to be applicable to longer duration as well.

Currently, the operators of MSN only allow us to inject /32 prefixes into the network in order to restrict the impact of Entact on customer traffic. As a result, we have limited capability in implementing a non-default TE strategy since we cannot arbitrarily change the DC selection or route selection for any prefix. Instead, we can only simulate a non-default TE strategy based on the routing, performance and traffic data collected under the default TE strategy in MSN. When presenting the following TE optimization results, we assume that the routing, performance and traffic to each prefix do not change under different TE strategies. This is a common assumption made by most of the existing work on TE [9, 12, 15]. We hope to study the effectiveness of Entact without such restrictions in the future.

7.1 Benefits of TE optimization

Figure 9 compares the wRTT and cost of four TE strategies, including the default, Entact$_{10}$ ($K = 10$), LowestCost (minimizing cost with $K = 0$), and BestPerf (minimizing wRTT with $K = inf$). We use 20-minute TE

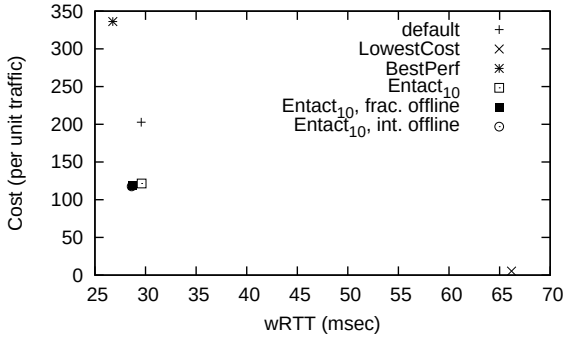

Figure 9: Comparison of various TE strategies.

window and 4 alternative routes from each DC for TE optimization. The x-axis is the wRTT in milliseconds and the y-axis is the relative cost. We cannot reveal the actual dollar cost for confidentiality reason. There is a big gap between the default strategy and Entact$_{10}$, which indicates the former is far from optimal. In fact, Entact$_{10}$ can reduce the default cost by 40% without inflating wRTT. This could lead to enormous amount of savings for MSN since it spends tens of millions of dollars a year on transit traffic cost.

We also notice there is significant tradeoff between cost and performance among the optimal strategies. In one extreme, the LowestCost strategy can eliminate almost all the transit cost by diverting traffic to free peering links. But this comes at the expense of inflating the default wRTT by 38 ms. Such a large RTT increase will notably degrade user-perceived performance when amplified by the many round trips involved in downloading content-rich Web pages. In the other extreme, the BestPerf strategy can reduce the default wRTT by 3 ms while increasing the default cost by 66%. This is not an appropriate strategy either given the relatively large cost increase and small performance gain. Entact$_{10}$ appears to be at a "sweet-spot" between the two extremes. By exposing the performance and cost of various opti-

path type	prefix	wRTT (ms)	pseudo cost
same	88.2%	29.6	41.1
pricier, longer	0.1%	74.5→75.1	195.1→195.1
pricier, shorter	4.6%	44.7→30.2	13.7→55.8
cheaper, longer	5.5%	27.6→39.8	738.3→177.8
cheaper, shorter	1.7%	55.5→47.8	483.7→174.4

Table 2: Comparison of paths under the default and Entact$_{10}$ strategies in terms of performance and cost.

path type	prefix
non-default DC, default route	2.1%
non-default DC, non-default route	2.5%
default DC, non-default route	7.2%

Table 3: Comparison of paths under the default and Entact$_{10}$ strategies in terms of DC selection and route selection.

mal strategies, the operators can make a more informed decision regarding which is a desirable operating point.

To better understand the source of the improvement offered by Entact$_{10}$, we compare Entact$_{10}$ with the default strategy during a typical 20-minute TE window. Table 2 breaks down the prefixes based on their relative pseudo cost and performance under these two strategies. Overall, the majority (88.2%) of the prefixes are assigned to the default path in Entact$_{10}$. Among the remaining prefixes, very few (0.1%) use a non-default path that is both longer and pricier than the default path (which is well expected). Only a small number of prefixes (1.7%) use a non-default path that is both cheaper and shorter. In contrast, 10.1% of the prefixes use a non-default path that is better in one metric but worse in the other. This means Entact$_{10}$ is actually making some "intelligent" performance-cost tradeoff for different prefixes instead of simply assigning each prefix to a "better" non-default path. For instance, 4.6% of the prefixes use a shorter but pricier non-default path. While this slightly increases the pseudo cost by 42.1, it helps to reduce the wRTT of these prefixes by 14.5 ms. More importantly, it frees up the capacity on some cheap peering links which can be used to carry traffic for certain prefixes that incur high pseudo cost under the default strategy. 5.5% of the prefixes use a cheaper but longer non-default path. This helps to drastically cut the pseudo cost by 560.5 at the expense of a moderate increase of wRTT (12.2 ms) for these prefixes. Note that Entact$_{10}$ may not find a free path for every prefix due to the performance and capacity constraints. The complexity of the TE strategy within each TE window and the dynamics of TE optimization across time underscore the importance of employing an automated TE scheme like Entact in a large OSP.

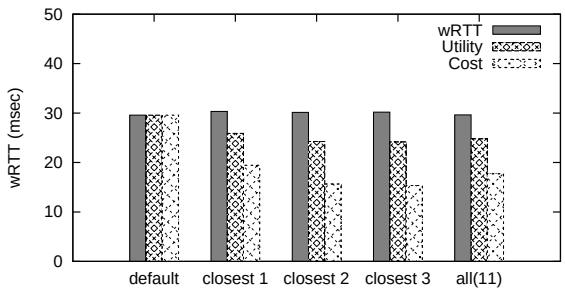

Figure 10: Effect of DC selection on TE optimization. (Utility and cost are scaled according to wRTT of the default strategy.)

Table 3 breaks down the prefixes that use a non-default path under Entact$_{10}$ during the 20-minute TE window by whether a non-default DC or a non-default route from a DC is used. Both non-default DCs and non-default routes are used under Entact$_{10}$ — 4.6% of the prefixes use a non-default DC and 9.7% of them use a non-default route from a DC. Non-default routes appear to be more important than non-default DCs in TE optimization. We will further study the effect of DC selection and route selection in §7.2 and §7.3.

Figure 9 shows that the difference between the integral and fractional solutions of Entact$_{10}$ is negligibly small. In TE optimization, the traffic to a prefix will be split across multiple alternative paths only when some alternative paths do not have enough capacity to accommodate all the traffic to that prefix. This seldom happens because the traffic volume to a prefix is relatively small compared to the capacity of a peering link in MSN.

We also compare the online Entact$_{10}$ with the offline one. In the latter case, we directly use the routing, performance, and traffic volume information of a 20-minute TE window to optimize TE in the same window. This represents the ideal case where there is no prediction error. Figure 9 shows the online Entact$_{10}$ incurs only a little extra wRTT and cost compared to the offline one (The two strategy points almost completely overlap). This is because the RTT and traffic to most of the prefixes are quite stable during such a short period (*e.g.*, 20 minutes). We will study to what extent the TE window affects the optimization results in §7.4.

7.2 Effects of DC selection

We now study the effects of DC selection on TE optimization. A larger number of DCs will provide more alternative paths for TE optimization, which in turn should lead to better improvement over the default strategy.

Nonetheless, this will also incur greater overhead in RTT measurement and TE optimization. We want to understand how many DCs are required to attain most of the TE optimization benefits. For each prefix, we sort the 11 DCs based on the RTT of the default route from each DC. We only use the RTT measurements taken in the first TE window of the evaluation period to sort the DCs. The ordering of the DCs should be quite stable and can be updated at a coarse-granularity, *e.g.*, once a week. We develop a slightly modified Entact^n_k which only considers the alternative paths from the closest n DCs to each prefix for TE optimization.

Figure 10 compares the wRTT, cost, and utility (§4.2.2) of Entact$^n_{10}$ as n varies from 1 to 11. We use 4 alternative routes from each DC to each prefix. Given a TE window, as n changes, the optimal strategy curve and the optimal strategy selected by Entact$^n_{10}$ will change accordingly. This complicates the comparison between two different Entact$^n_{10}$'s since one of them may have higher cost but smaller wRTT. For this reason, we focus on comparing the utility for different values of n. As shown in the figure, Entact$^1_{10}$ (only with route selection but no DC selection) and Entact$^2_{10}$ can cut the utility by 12% and 18% respectively compared to the default strategy. The utility reduction diminishes as n exceeds 2. This suggests that TE optimization benefits can be attributed to both route selection and DC selection. Moreover, selecting the closest two DCs for each prefix seems to attain almost all the TE optimization benefits. Further investigation reveals that most prefixes have at most two nearby DCs. Using more DCs generally will not help TE optimization because the RTT from those DCs is too large.

Note that the utility of Entact$^{11}_{10}$ is slightly higher than that of Entact$^2_{10}$. This is because the utility of Entact^n_k is computed from the 95% traffic cost during the entire evaluation period. However, Entact^n_k only minimizes pseudo utility computed from pseudo cost in each TE window. Even though the pseudo utility obtained by Entact^n_k in a TE window always decreases as n grows, the utility over the entire evaluation period may actually move in the opposite direction.

7.3 Effects of alternative routes

We evaluate how TE optimization is affected by the number of alternative routes (m) from each DC. A larger m will not only offer more flexibility in TE optimization but also incur greater overhead in terms of route injection, optimization, and RTT measurement. In this experiment, we measure the RTT of 8 alternative routes from each DC to each prefix every 20 minutes for 1 day. Figure 11 illustrates the wRTT, cost, and utility of Entact$_{10}$ under different m. For the same reason as in the previous section, we focus on comparing utility. As m grows from 1 to 3, the utility gradually decreases up to 14% compared to the default strategy. The utility almost remains the same after m exceeds 3. This suggests that 2 to 3 alternative routes are sufficient for TE optimization in MSN.

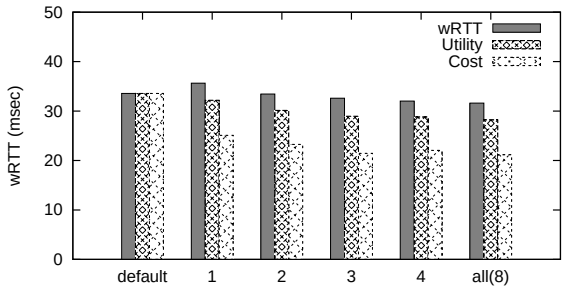

Figure 11: Effect of the number of alternative routes on TE optimization.

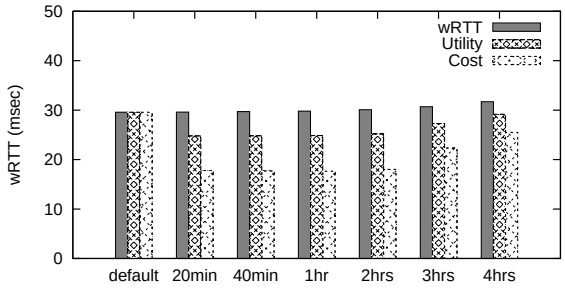

Figure 12: Effect of the TE window on TE optimization.

7.4 Effects of TE window

Finally, we study the impact of TE window on optimization results. Entact performs online TE in a TE window using predicted performance and traffic information (§5). On the one hand, both performance and traffic volume can vary significantly within a large TE window. It will be extremely difficult to find a fixed TE strategy that performs well during the entire TE window. On the other hand, a small TE window will incur high overhead in route injection, RTT measurement, and TE optimization. It may even lead to frequent user-perceived performance variations.

Figure 12 illustrates the wRTT, cost, and utility of Entact$_{10}$ under different TE window sizes from 20 minutes to 4 hours. As before, we focus on comparing the utility. We still use 4 alternative routes from each DC to each prefix. Entact$_{10}$ can attain about the same utility reduction compared to the default strategy when the TE

# routes	injection time (sec)	CPU (%)	RIB (MB)	FIB (MB)
5,000	9	3	0.81	0.99
10,000	15	2	1.61	1.72
20,000	30	3	3.22	3.18
30,000	51	4	4.84	4.64
50,000	73	7	8.06	7.57
100,000	147	17	16.12	14.88

Table 4: Route injection overhead measured on a testbed.

window is under 1 hour. This is because the performance and traffic volume are relatively stable during such time scale. As the TE window exceeds 1 hour, the utility noticeably increases. With a 4-hour TE window, Entact$_{10}$ can only reduce the default utility by 1%. In fact, because the traffic volume can fluctuate over a wide range during 4 hours, Entact$_{10}$ effectively optimizes TE for the peak interval to avoid link congestion. This leads to a sub-optimal TE strategy for many non-peak intervals. In §8, we show that an 1-hour TE window imposes reasonably low overhead.

8 Online TE Optimization Overhead

So far, we have demonstrated the benefits provided by Entact. In this section, we study the feasibility of deploying Entact to perform full-scale online TE optimization in a large OSP. The key factor that determines the overheads of Entact is the number of prefixes. While there are roughly 300K Internet prefixes in total, we will focus on the top 30K high-volume prefixes that account for 90% of the traffic in MSN (§6.1). Multi-location prefixes may inflate the actual number of prefixes beyond 30K; we leave the study of multi-location prefixes as future work. The results in §7.3 and §7.4 suggest that Entact can attain most of the benefits by using 2 alternative routes from each DC and an 1-hour TE window. We now evaluate the performance and scalability of key Entact components under these settings.

8.1 Route injection

We evaluate the route injection overhead by setting up a router testbed in the Schooner lab [4]. The testbed comprises a Cisco 12000 router and a PC running our route injector. Cisco 12000 routers are commonly used in the backbone network of large OSPs. When Entact initializes, it needs to inject 30K routes into each router in order to measure the RTT of the default route and one non-default route simultaneously. This injection process can be spread over several days to avoid overloading routers. Table 4 shows the size of the RIB (routing information base) and FIB (forwarding information base) as the number of injected routes grows. 30K routes merely occupy about 4.8 MB in the RIB and FIB. Such memory overhead is relatively small given that today's routers typically hold roughly 300K routes (the number of all Internet prefixes).

After the initial injection is done, Entact needs to continually inject routes to apply the output of the online TE optimization. Table 4 also shows the injection time of different number of routes. It takes only 51 seconds to inject 30K routes, which is negligibly small compared to the 1-hour TE window. We expect the actual number of injected routes in a TE window to be much smaller because most prefixes will simply use a default route (§7.1).

8.2 Impact on downstream ISPs

Compared to the default TE strategy, the online TE optimization performed by Entact may cause traffic to shift more frequently. This is because Entact needs to continually adapt to changes in performance and traffic volume in an OSP. A large traffic shift may even overload certain links in downstream ISPs, raising challenges in the TE of these downstream ISPs. This problem may exacerbate if multiple large OSPs perform such online TE optimization simultaneously. Given a 5-minute interval i, we define a *total traffic shift* to quantify the impact of an online TE strategy on downstream ISPs:

$$TotalShift_i = \sum_p shift_i(p) / \sum_p vol_i(p)$$

Here, $vol_i(p)$ is the traffic volume to prefix p and $shift_i(p)$ is the traffic shift to p in interval i. If p stays on the same path in intervals i and $i-1$, $shift_i(p)$ is computed as the increase of $vol_i(p)$ over $vol_{i-1}(p)$. Otherwise, $shift_i(p) = vol_i(p)$. In essence, $shift_i(p)$ captures the additional traffic load imposed on downstream ISPs relative to the previous interval. The additional traffic load is either due to path change or due to natural traffic demand growth.

Figure 13 compares the $TotalShift$ under the static TE strategy, the default TE strategy, and Entact$_{10}$ over the entire evaluation period. In the static strategy, the TE remains the same across different intervals, and its traffic shift is entirely caused by natural traffic demand variations. We observe that most of the traffic shift is actually caused by natural traffic demand variations. The traffic shift of Entact$_{10}$ is only slightly larger than that of the default strategy. As explained in §7.1, Entact$_{10}$ assigns a majority of the prefixes to a default path and only reshuffles the traffic to roughly 10% of the prefixes. Moreover, the paths of those 10% prefixes do not always

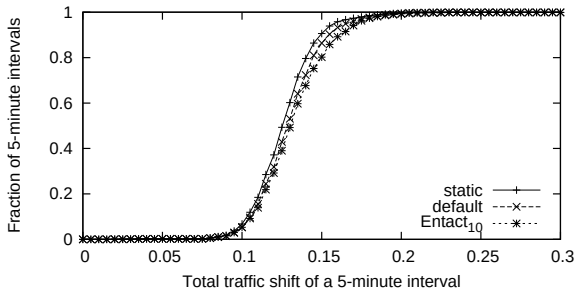

Figure 13: Traffic shift under the static, default, and Entact$_{10}$ TE strategies

change across different intervals. As a result, Entact$_{10}$ incurs limited extra traffic shift compared to the default strategy.

8.3 Computation time

Entact$_k$ computes an optimal strategy in two steps: i) solving an LP problem to find a fractional solution; ii) converting the fractional solution into an integer one. Let n be the number of prefixes, d be the number of DCs, and l be the number of peering links. The number of variables f_{ijk} in the LP problem is $n \times d \times l$. Since d and l are usually much smaller than n and do not grow with n, we consider the size of the LP problem to be $O(n)$. The worst case complexity of an LP problem is $O(n^{3.5})$ [1]. The heuristic for converting the fractional solution into an integer one (§4.2.3) requires n iterations to assign n prefixes. In each iteration, it takes $O(n \log(n))$ to sort the unassigned prefixes in the worst case. Therefore, the complexity of this step is $O(n^2 \log(n))$.

We evaluate the time to solve the LP problem since it is the computation bottleneck in TE optimization. We use Mosek [6] as the LP solver and measure the optimization time of one TE window on a Windows Server 2008 machine with two 2.5 GHz Xeon processors and 16 GB memory. We run two experiments using the top 20K high-volume prefixes and all the 300K prefixes respectively. The RTTs of the 20K prefixes are from real measurement while the RTTs of the 300K prefixes are generated based on the RTT distribution of the 20K prefixes. We consider 2 alternative routes from each of the 11 DCs to each prefix. The traffic volume, routing, and link price and capacity information are directly drawn from the MSN dataset. The running time of the two experiments are 9 and 171 seconds respectively, representing a small fraction of an 1-hour TE window.

8.4 Probing requirement

To probe 30K prefixes in an 1-hour TE window, the bandwidth usage of each prober will be 30K (prefixes) x 2 (alternative routes) x 2 (RTT measurements) x 5 (TCP packets) x 80 (bytes) / 3600 (seconds) = 0.1 Mbps. Such overhead is negligibly small.

8.5 Processing traffic data

We use a Windows Server 2008 machine with two 2.5 GHz Xeon processors and 16 GB memory to collect and process the Netflow data from all the routers in MSN. It takes about 80 seconds to process the traffic data of one 5-minute interval during peak time. Because Netflow data is processed on-the-fly as the data is streamed to Entact, such processing speed is fast enough for online TE optimization.

9 Related Work

Our work is closely related to the recent work on exploring route diversity in multihoming, which broadly falls into two categories. The first category includes measurement studies that aim to quantify the potential performance benefits of exploring route diversity, including the comparative study of overlay routing *vs.* multihoming [7,8,11,24]. These studies typically ignore the cost of the multihoming connectivity. In [7], Akella *et al.* quantify the potential performance benefits of multihoming using traces collected from a large CDN network. Their results show that smart route selection has the potential to achieve an average performance improvement of 25% or more for a 2-multihomed customer in most cases, and most of the benefits of multihoming can be achieved using 4 providers. Our work differs from these studies in that it considers both performance and cost.

The second category of work on multihoming includes algorithmic studies of route selection to optimize cost, or performance under certain cost constraint [12, 15]. For example, Goldenberg *et al.* [15] design a number of algorithms that assign individual flows to multiple providers to optimize the total cost or the total latency for all the flows under fixed cost constraint. Dhamdhere and Dovrolis [12] develop algorithms for selecting ISPs for multihoming to minimize cost and maximize availability, and for egress route selection that minimizes the total cost under the constraint of no congestion. Our work differs from these algorithmic studies in a few major ways. First, we propose a novel joint TE optimization technique that searches for the optimal "sweet-spot" in the performance-cost continuum. Second, we present the design and implementation details of a route-injection-

based technique that measures the performance of alternative paths in real-time. Finally, to our knowledge, we provide the first TE study on a large OSP network which exhibits significantly different characteristics from multihoming stub networks previously studied.

Our work as well as previous work on route selection in multihoming differ from numerous work on intra- and inter-domain traffic engineering, *e.g.*, [13, 18, 23]. The focus of these later studies is on balancing the utilization of ISP links instead of on optimizing end-to-end user performance.

10 Conclusion

We studied the problem of optimizing cost and performance of carrying traffic for an OSP network. This problem is unique in that an OSP has the flexibility to source traffic from different data centers around the globe and has hundreds of connections to ISPs, many of which carry traffic to only parts of the Internet. We formulated the TE optimization problem in OSP networks, and presented the design of the Entact online TE scheme. Using our prototype implementation, we conducted a trace-driven evaluation of Entact for a large OSP with 11 data centers. We found that that Entact can help this OSP reduce the traffic cost by 40% without compromising performance. We also found these benefits can be realized with acceptably low overheads.

11 Acknowledgments

Murtaza Motiwala and Marek Jedrzejewicz helped with collecting Netflow data. Iona Yuan and Mark Kasten helped with maintaining BGP sessions with the routers in MSN. We thank them all.

We also thank Jennifer Rexford for shepherding this paper and the NSDI 2010 reviewers for their feedback.

References

[1] Linear programming. `http://en.wikipedia.org/wiki/Linear_programming`. Retrieved March 2010.

[2] Nielsen Online. `http://www.nielsen-online.com/`.

[3] Quagga Routing Suite. `http://www.quagga.net/`.

[4] Schooner User-Configurable Lab Environment. `http://www.schooner.wail.wisc.edu/`.

[5] Share your favorite personal Windows Live Messenger story with the world! `http://messengersays.spaces.live.com/Blog/cns!5B410F7FD930829E!73591.en%try`. Retrieved March 2010.

[6] The MOSEK Optimization Software. `http://www.mosek.com/`.

[7] AKELLA, A., MAGGS, B., SESHAN, S., SHAIKH, A., AND SITARAMAN, R. A Measurement-Based Analysis of Multihoming. In *Proc. of ACM SIGCOMM* (2003).

[8] AKELLA, A., PANG, J., MAGGS, B., SESHAN, S., AND SHAIKH, A. A Comparison of Overlay Routing and Multihoming Route Control. In *Proc. of ACM SIGCOMM* (2004).

[9] CAO, Z., WANG, Z., AND ZEGURA, E. Performance of hashing-based schemes for Internet load balancing. In *Proc. of IEEE INFOCOM* (2001).

[10] DABEK, F., COX, R., KAASHOEK, F., AND MORRIS, R. Vivaldi: A Decentralized Network Coordinate System. In *Proc. of ACM SIGCOMM* (2004).

[11] DAI, R., STAHL, D. O., AND WHINSTON, A. B. The Economics of Smart Routing and QoS. In *Proc. of the 5th International Workshop on Networked Group Communications (NGC)* (2003).

[12] DHAMDHERE, A., AND DOVROLIS, C. ISP and Egress Path Selection for Multihomed Networks. In *Proc. of IEEE INFOCOM* (2006).

[13] FEAMSTER, N., BORKENHAGEN, J., AND REXFORD, J. Guidelines for interdomain traffic engineering. *ACM SIGCOMM Computer Communications Review* (2003).

[14] FRANCIS, P., JAMIN, S., JIN, C., JIN, Y., RAZ, D., SHAVITT, Y., AND ZHANG, L. IDMaps: A Global Internet Host Distance Estimation Service. *IEEE/ACM Transactions on Networking* (2001).

[15] GOLDENBERG, D., QIU, L., XIE, H., YANG, Y. R., AND ZHANG, Y. Optimizing Cost and Performance for Multihoming. In *Proc. of ACM SIGCOMM* (2004).

[16] GUMMADI, K. P., MADHYASTHA, H., GRIBBLE, S. D., LEVY, H. M., AND WETHERALL, D. J. Improving the Reliability of Internet Paths with One-hop Source Routing. In *Proc. of OSDI* (2004).

[17] HU, N., AND STEENKISTE, P. Evaluation and characterization of available bandwidth probing techniques. *IEEE JSAC Special Issue in Internet and WWW Measurement, Mapping, and Modeling* (2003).

[18] IETF TRAFFIC ENGINEERING WORKING GROUP. `http://www.ietf.org/htmk.chaters/tewg-charter.html`.

[19] JAIN, M., AND DOVROLIS, C. End-to-end available bandwidth: Measurement methodology, dynamics, and relation with tcp throughput. In *Proc. of ACM SIGCOMM* (2002).

[20] KOHAVI, R., HENNE, R. M., AND SOMMERFIELD, D. Practical guide to controlled experiments on the web: Listen to your customers not to the hippo. In *Proc. of SIGKDD* (2007).

[21] MYTHILI, M. F., VUTUKURU, M. J. F. M., FEAMSTER, N., AND BALAKRISHNAN, H. Geographic locality of ip prefixes. In *Proc. of the Internet Measurement Conference (IMC)* (2005).

[22] NG, T. S. E., AND ZHANG, H. Predicting Internet Network Distance with Coordinates-Based Approaches. In *Proc. of IEEE INFOCOM* (2002).

[23] ROUGHAN, M., THORUP, M., AND ZHANG, Y. Traffic Engineering with Estimated Traffic Matrices. In *Proc. of the Internet Measurement Conference (IMC)* (2003).

[24] SEVCIK, P., AND BARTLETT, J. Improving User Experience with Route Control. Tech. Rep. 5062, NetForecast Inc., 2002.

[25] STRAUSS, J., KATABI, D., KAASHOEK, F., AND PRABHAKAR, B. Spruce: A lightweight end-to-end tool for measuring available bandwidth. In *Proc. of the Internet Measurement Conference (IMC)* (2003).

[26] VAZIRANI, V. V. *Approximation Algorithms*. Springer-Verlag, 2001.

[27] WONG, B., SLIVKINS, A., AND SIRER, E. G. Meridian: a lightweight network location service without virtual coordinates. In *Proc. of ACM SIGCOMM* (2005).

[28] ZEITOUN, A., AND JAMIN, S. Rapid Exploration of Internet Live Address Space Using Optimal Discovery Path. In *Proc. of IEEE Globecom* (2003).