Traffic Engineering With Traditional IP Routing Protocols

Bernard Fortz
Institut d'Administration et de Gestion
Universite Catholique de Louvain
Louvain-la-Neuve, Belgium
fortz@poms.ucl.ac.be

Jennifer Rexford Mikkel Thorup
Internet and Networking Systems
AT&T Labs – Research
Florham Park, NJ 07932
{jrex,mthorup}@research.att.com

Abstract

Traffic engineering involves adapting the routing of traffic to the network conditions, with the joint goals of good user performance and efficient use of network resources. In this paper, we describe an approach to intradomain traffic engineering that works within the existing deployed base of Interior Gateway Protocols (IGPs), such as Open Shortest Path First (OSPF) and Intermediate System-Intermediate System (IS-IS). We explain how to adapt the configuration of link weights, based on a network-wide view of the traffic and topology within a domain. In addition, we summarize the results of several studies of techniques for optimizing OSPF/IS-IS weights to the prevailing traffic. The paper argues that conventional IGPs are surprisingly effective for engineering the flow of traffic in large IP networks.

1 Introduction

In some sense, IP networks manage themselves. A host implementing the Transmission Control Protocol (TCP) adjusts its sending rate to the bandwidth available on the path to the destination, and routers react to changes in the network topology by computing new paths. This has made the Internet an extremely robust communication network, even in the face of rapid growth and occasional failures. However, these mechanisms do not ensure that the network runs *efficiently*. For example, a particular link may be congested despite the presence of under-utilized links in other parts of the network. Or, a voice-over-IP call may travel over a route with high propagation delay when a low-latency path is available. Improving user performance and making more efficient use of network resources requires adapting the routing of traffic to the prevailing demands. This task is referred to as *traffic engineering* [1, 2]. Most work on traffic engineering has focused on techniques for controlling the flow of traffic within a single Autonomous System (AS), such as a company, university campus, or Internet Service Provider (ISP).

1.1 Intradomain Traffic Engineering

Traffic engineering depends on having a set of performance objectives that guide the selection of paths, as well as effective mechanisms for the routers to select paths that satisfy these objectives. Most existing IP

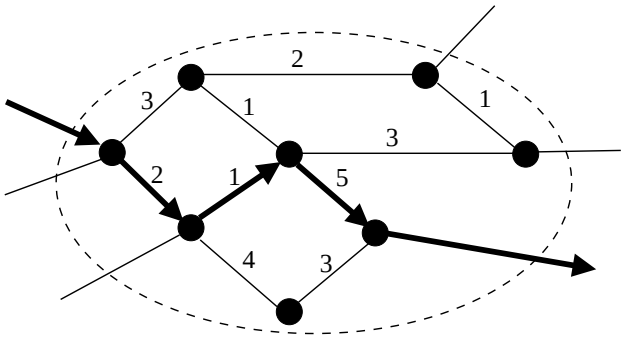
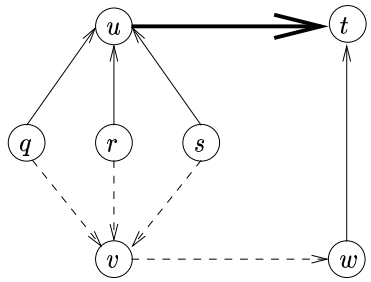


Figure 1: Shortest path routing within an Autonomous System based on OSPF/IS-IS link weights

networks run Interior Gateway Protocols (IGPs) such as OSPF (Open Shortest Path First) or IS-IS (Intermediate System-Intermediate System) that select paths based on static link weights configured by network operators. Routers use these protocols to exchange link weights and construct a complete view of the topology inside the AS, as shown in Figure 1. Then, each router computes shortest paths (as the sum of these weights) and creates a table that controls the forwarding of each IP packet to the next hop in its route. In this paper, we focus on the techniques for *selecting* the paths rather than the underlying mechanisms for packet forwarding. Traditionally, IP forwarding depends on the destination address in the IP header of each packet. More recently, routers running Multi-Protocol Label Switching (MPLS) can forward packets based on the label in the MPLS header. In either case, we are concerned with how the path is chosen rather than how the packets are forwarded.

On the surface, the basic framework of shortest-path routing does not seem flexible enough to support traffic engineering in an IP network supporting a diverse set of applications. First of all, these IGPs are limited to routing scenarios that can be specified with a single integer weight on each link. However, we argue that link weights suffice to specify near-optimal routing for large, real-world networks. Second of all, in their basic forms, the OSPF and IS-IS protocols do not adapt the link weights in response to changes in traffic or the failures of network elements, and the path-selection process does not directly incorporate any performance objectives (beyond the selection of a "shortest" path). Recent standards activity has proposed traffic-engineering extensions to OSPF and IS-IS to incorporate information about the prevailing traffic into the link-state advertisements and the path selection decisions [3, 4]. However, these extensions require modifications to the routers to collect and disseminate information about network load and compute and establish paths based on the load metrics. Instead, we argue that it is often possible to select *static* link weights that are resilient to traffic fluctuations and link failures, allowing the use of the traditional incarnations of OSPF and IS-IS.

The example in Figure 2 from [5] shows how to control the distribution of traffic in a network by



Link	Default unit weights		Increasing weight of overloaded link		Optimal global single change	
	Weight	Load	Weight	Load	Weight	Load
(q, u)	1	1	1	0.5	1	1
(r, u)	1	1	1	0.5	1	1
(s, u)	1	1	1	0.5	3	0
(u, t)	1	**3**	2	1.5	1	**2**
(q, v)	1	0	1	0.5	1	0
(r, v)	1	0	1	0.5	1	0
(s, v)	1	0	1	0.5	1	1
(v, w)	1	0	1	1.5	1	1
(w, t)	1	1	1	**2.5**	1	**2**

Figure 2: Small example of global optimization of IGP link weights

tuning the IGP weights. The left side of the figure shows an example network where all links have the same capacity and each of the nodes q, r, s, and w have one unit of traffic to send to node t. The simple performance objective here is to minimize the maximum link load. The table on the right side of the figure shows the load on each link for certain weight settings:

- **Initial configuration with unit weights:** The first column in the table shows the results of having the same weight of 1 on every link. This directs all of the traffic from nodes q, r, and s through node u, forcing three units of load on link (u, t).
- **Local change to the weight of the congested link:** A naive, local approach to reducing the load might increase the IGP weight on the overloaded link (u, t) to 2. This configuration results in two shortest paths for nodes q, r, and s, and a splitting of traffic over paths via u and v. The second column in the table shows that this setting of the weights places a heavy load on the link (w, t).
- **Global optimization of the link weights:** A global optimization of the weights would produce a weight setting like the one in the third column, with no link carrying more than two units of traffic. For this particular example, the distribution of traffic is optimal with regard to the maximum load. Since 4 units of traffic have to reach node t along its two incoming edges, no other routing scheme could produce a better solution.

In this example, changing the link weights to alleviate the congestion on the link (u, t) is an attractive alternative to buying and deploying additional bandwidth between routers u and t.

1.2 Advantages of Using Traditional IGPs

This paper presents an overview of a practical approach to working within the existing framework of static link weights, without modification to the routing protocols or the routers themselves. The paper brings together the work in various papers that describe individual components of this approach to traffic engineering.

The main point underlying this body of work is that the process of arriving at good values for the weights, or a good set of changes to the existing values of the weights, is handled *externally* from the routers. This process could depend on traffic measurements and topology data collected from the operational network. The selection of the weights may also depend on a wide variety of different cost, performance, and reliability constraints. The link weights are configured by an external entity, such as a network management system or a human operator, to achieve certain traffic engineering goals. Generally, we view a modification of the link weights as a significant change to the network that is performed on a relatively coarse time scale.

This approach has the advantage enabling operators to engineer the flow of traffic in their networks while retaining the simplicity of existing IGPs. The proposed approach has two key features—a centralized approach to setting the routing parameters and the use of link weights as the way to drive the path-selection process. The approach is centralized in that the routing parameters are set based on a network-wide view of the topology and traffic, rather than the local views at each router. This centralized approach has several advantages, as follows:

- **Protocol stability:** The routers in the network do not adapt automatically to locally-constructed (potentially out-of-date) views of the traffic. The paths do not change unless the routing parameters are reconfigured or the network topology changes. This predictability aids network operators in diagnosing performance problems.
- **Low protocol overhead:** The routers inside the network do not need to track changes in load and disseminate new link-state information. This limits the bandwidth consumed by the routing protocol and the computational load imposed on the routers.
- **Diverse performance constraints:** The process for selecting the routing parameters can depend on a wide variety of performance and reliability constraints. New constraints and improved techniques for selecting the routing parameters can be incorporated without changes to the routing protocols or the router implementations. In addition, operators can incorporate constraints that are difficult to formalize in a routing protocol.

Using link weights to express the routing configuration has the following advantages:

- **Compatibility with traditional IGPs:** Selecting the link weights outside of the network allows operators to engineer the flow of traffic while working with the existing OSPF and IS-IS routing protocols. This avoids the need to upgrade equipment or introduce additional configuration complexity.
- **Concise representation:** Link weights are a concise form of configuration state. Each router is configured with the weight (and perhaps area) for each of its outgoing links. The router does not need to be configured with any path-level information or with any state concerning the incident edges at other routers. In addition, an operator can change multiple paths in the network with the change of a single link weight.

- **Default weights and backup routes:** Link weights can have a reasonable default configuration based on link capacity (e.g., inversely proportional to capacity). If the topology changes (e.g., a link failure), the router can automatically compute new routes based on the current topology and link weights. These routes can carry traffic until a new configuration is selected.

Despite these advantages, traffic engineering using conventional IGPs requires overcoming several practical challenges. In Section 2, we outline the key components of a system for assigning link weights based on the traffic demands, network topology, and performance objectives. Next, in Section 3 we discuss the effectiveness of shortest-path routing based on link weights in providing control over the flow of traffic in the network; discussion of the algorithm we use to optimize the link weights is deferred to the Appendix. Section 4 concludes the paper with a discussion of future research directions.

2 Traffic Engineering Framework

In this section, we formalize an approach to traffic engineering based on external changes in the IGP configuration. Assigning link weights based on the traffic demands and performance objectives depends on several key ingredients, as illustrated in Figure 3. First, instrumentation of the operational network should provide information about the status of the network elements and the current offered traffic. In practice, this topology and traffic data are necessary for a variety of other operational tasks. Second, evaluating possible settings of the link weights depends on having an accurate model of how the IGP configuration affects the flow of traffic. Third, selecting good settings of the weights depends on having an objective function that captures the key performance and reliability constraints, as well as an efficient algorithm for computing weights that satisfy these constraints; we discuss these optimization issues in more detail in Section 3. Fourth, after deciding on the values of the weights, an automated system or a human operator needs to effect these changes in the operational network.

2.1 Inputs: Topology and Configuration

Selecting good link weights depends on having a timely and accurate view of the current state of the network. This view includes the operational routers and links in the network, as well as the capacity of the links and the current configuration of the IGP parameters (e.g., OSPF/IS-IS weight and area). Tracking this information is important for a variety of operational purposes. For example, a Network Operations Center (NOC) may display the current topology on a wallboard to track the failures of network elements or changes in basic statistics (e.g., router CPU load and link utilization).

Topology and configuration information are available from a variety of sources. Link capacity and IGP parameters are available from router configuration data (such as configuration files) [6] and may also

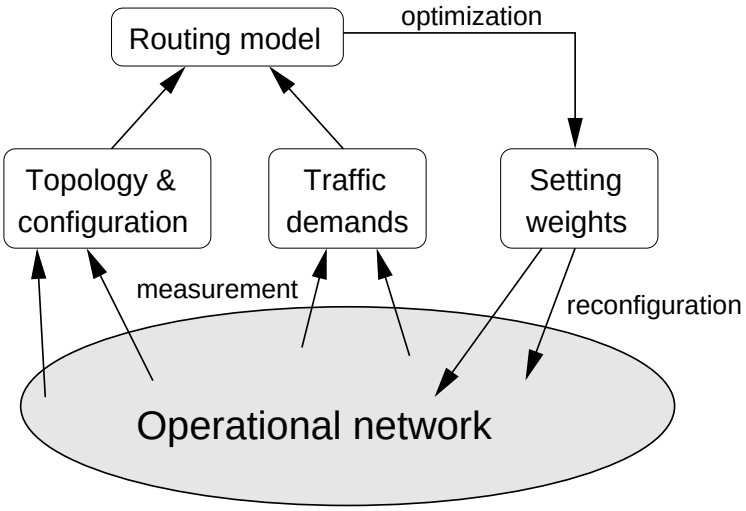


Figure 3: Key components of the traffic engineering framework

be stored in external databases that drive the provisioning of the network elements. The Simple Network Management Protocol (SNMP) provides information about the status of the network elements, either by polling or via traps. In addition, it is possible to deploy IGP route monitors that track the topology and IGP parameters in the operational network. For example, a software router that participates in OSPF/IS-IS with the operational routers could track and report this information in real time [7].

2.2 Inputs: Traffic Demands

The selection of weights depends on having an estimate of the offered load on the network, in terms of the volume of traffic between each pair of routers or each pair of edge links. This kind of information is necessary for designing the network and planning the outlay of new capacity. In some cases, the operator may have an estimate of the traffic demand based on past experience or customer subscription information, such as the amount of bandwidth allocated for a Virtual Leased Line or Virtual Private Network service. In other cases, the traffic demands can be gleaned by measuring the traffic in the operational network. Computing estimates of the offered load requires combining measurement data from multiple locations in the network to compute the traffic demands. This instrumentation and collection infrastructure may be necessary to support other operational tasks, such as basic usage reporting for billing, marketing, and capacity planning.

Constructing a network-wide view of the traffic demands requires relatively sophisticated techniques for the collection and analysis of measurement data. Four main approaches have been considered, which are described in more detail in [8]. First, the necessary traffic statistics may be available directly from SNMP

Management Information Bases (MIBs), depending on the forwarding paradigm employed in the network. For example, MPLS MIBs were used to measure the volume of traffic on the Label Switched Path (LSP) between each pair of edge routers in Global Crossing's backbone [9]. Second, the offered traffic can be computed by combining packet-level or flow-level measurements at the network edge with the information available in routing tables. This approach has been applied in the AT&T and Sprint networks [10, 11]. Third, if fine-grain traffic measurements are not available, the offered traffic may be inferred based on observations of the aggregate load on links inside the network in conjunction with routing data. This approach has been applied in Lucent's IP network [12]. Fourth, new techniques for packet sampling offer the possibility of direct observation of the offered traffic as it flows through the network [13].

2.3 Routing Model

Traffic engineering requires an effective way to predict the flow of traffic through the network based on the routing configuration. Knowing the route(s) between each pair of nodes enables the operators to identify the traffic that imposes load on a congested link and evaluate the influence of possible changes to the IGP parameters. This requires an accurate model of how the routers in an AS compute paths based on the topology and IGP configuration. When all of the links belong to a single OSPF/IS-IS area, path selection simply involves computing the shortest path(s) between each pair of routers (e.g., using Dijkstra's algorithm). Larger networks are typically divided into multiple OSPF/IS-IS areas. For routers in different areas, the path selection depends on the summary information conveyed across area boundaries. In some cases, the network may have multiple shortest paths between the same pair of routers. The OSPF and IS-IS protocol specifications do not dictate how routers handle the presence of multiple shortest paths. In practice, most routers capitalize on the multiple paths to balance load. A router typically splits traffic roughly evenly[1] over each of the outgoing links along a shortest path to the destination, as shown in Figure 4.

Ultimately, then, the routing model should compute a *set* of paths for each pair of routers. These paths can be represented in terms of the fraction of the traffic (for this pair of routers) that traverses each of the links. The output of the routing model can be combined with the traffic demands to estimate the volume of traffic on each link, based on the topology and the IGP configuration. The routing model also plays a role in capturing the interaction of the IGP with interdomain routing (i.e., the Border Gateway Protocol (BGP)). A single block of destination IP addresses may be reachable via multiple exit points to neighboring domains. For example, an AS may have multiple links to another service provider at different geographic

[1] This is achieved by allowing an entry in the forwarding table to have multiple outgoing links. Rather than alternating between these links at the packet level, routers typically attempt to forward packets for the same source-destination pair along a single path; this reduces the likelihood that packets from the same TCP connection arrive out-of-order at the receiver. Load-balancing is typically achieved by performing a hash function on the source and destination IP addresses of each packet. The value of the hash function determines which of the outgoing links should carry the packet.

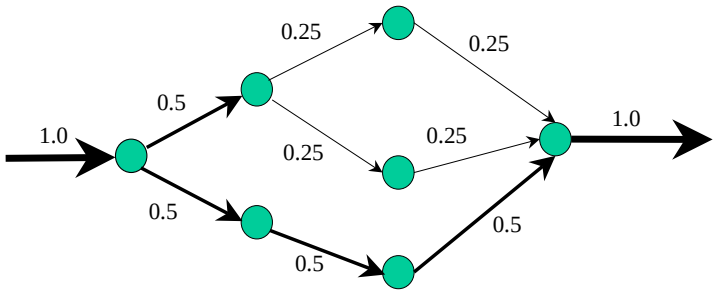


Figure 4: Traffic splitting across multiple shortest paths in the network

locations. The BGP decision process selects from these routes based on the IGP cost of the shortest path to each exit point [14, 15]. This enables each router to select the "closest" exit point. The work in [16] presents an overview of a routing model that captures the details of multiple OSPF/IS-IS areas, splitting over multiple shortest paths, and the influence of IGP parameters on how the traffic exits the network en route to a neighboring AS.

2.4 Output: Setting the Weights

Changing IGP weights requires applying the appropriate commands to the affected routers. This may involve running `telnet` or `ssh` to connect to each router's command-line interface. The specific commands depends on the particular operating system running on the router. For example, in Cisco IOS (Internet Operating System) parlance, the operator would enter a command like "`ip ospf cost 64`" in the context of the appropriate outgoing link to change the OSPF weight to 64. These commands may be applied manually by a network operator or performed automatically by a script. More generally, the service provider may have a network management system for configuring the routers. An integrated network management system could conceivably automate the entire process of detecting congestion, selecting suitable IGP weights, and effecting the configuration changes. However, given the complexity of operating a network, a service provider may have a human operator involved to oversee the process.

Following a weight change, the router updates its link-state database and floods the new value of the weight to the rest of the network. Upon receiving the new link-state advertisement, each router updates its link-state database, computes the new shortest paths, and updates certain entries in its forwarding table. During this convergence period, the routers in the network do not have a consistent view of the shortest-path routes for some destinations. This transient period is similar to what happens following a change in the topology of the network due to an equipment failure or the addition of a new router or link. Convergence following a weight change is typically faster than convergence after a failure, though, since the router does

not have to incur a delay to detect that a failure has occurred[2]. Still, changing a link weight does require the network to undergo a transient period where the forwarding paths are changing for some of the traffic. As such, we do not envision making frequent changes to the link weights. Instead, link weights would change under special circumstances following the outlay of new capacity, a significant equipment failure, or a serious shift in the traffic demands. Fortunately, a *single* setting of the link weights often suffices to handle the daily fluctuations of the traffic in a large IP backbone, as discussed in the next section.

3 Performance Properties

In this section, we discuss how we can engineer the flow of traffic using the traditional OSPF/IS-IS routing protocols in large networks, using an optimization algorithm to identify good IGP weight settings. First, we describe how to use objective functions to judge the quality of a particular solution to the routing problem. Next, we evaluate several heuristics for setting the link weights for a given topology and traffic matrix. Drawing on the experiments in [5], we show that good settings of the IGP weights perform within a few percent of an optimal distribution of traffic for realistic topologies and traffic demands. Then, we consider how to deal with fluctuations in the traffic demands over time without modifying the IGP weights and describe how to change the flow of traffic in the network with small modifications to the link weights. These results draw on the results of experiments in [19]. Readers interested in the results of experiments on a wide variety of real and synthetic topologies can refer to [20].

3.1 Objective Function

Any attempt to optimize the distribution of traffic depends on having an objective function that quantifies the "goodness" of a solution. The objective function typically considers the utilization of each link and biases against solutions that overload any part of the network. The simplest objective function is max-utilization [21], which measures the ratio of traffic load to capacity for the most heavily-loaded link in the network. We used this objective function earlier for our example in Figure 2 in Section 1. In our example, all links had the same capacity, so maximum utilization was equivalent to maximum load. Minimizing the max-utilization is a natural and intuitive objective for routing. However, this function is overly sensitive to individual bottleneck links that may be difficult to avoid. For example, an ingress link from a neighboring domain may carry a large amount of traffic under any routing solution. Considering the load on this link to the exclusion of the other traffic in the network does not necessarily result in the best solution. In addition,

[2] In some cases, a router must rely on heartbeat ("Hello") messages to detect that a link to a neighboring router has failed. Small heartbeat timers can reduce the latency for detecting failures [17, 18], at the expense of higher overhead for exchanging these messages.

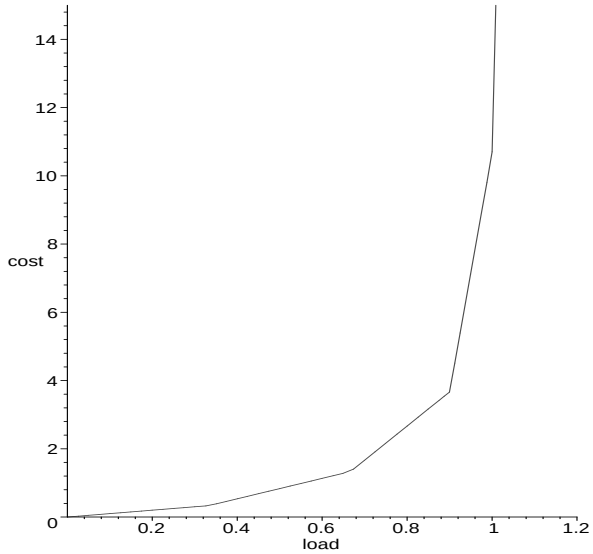


Figure 5: Link cost $\phi(\ell_a, 1)$ as a function of load ℓ_a with capacity $c_a = 1$

the max-utilization function does not penalize solutions that force traffic to traverse very long paths.

Instead, the objective function should consider the utilization of each of the links in the network, while still assigning a heavy penalty to solutions with one or more heavily-loaded links. To construct a network-wide view of the "goodness" of a routing solution, we consider objective functions of the form:

$$\Phi = \sum_{a \in A} \phi(\ell_a, c_a)$$

that sum a link cost $\phi(\ell_a, c_a)$ from each link a depending on the relation between the load ℓ_a and the capacity c_a. For example, the work in [22] uses a link cost of $\phi(\ell_a, c_a) = \ell_a/(c_a - \ell_a)$. The cost function attempts to estimate the queuing delay experienced by a packet traversing link a. The resulting objective function makes it progressively more expensive to send traffic along links as the load approaches the capacity. This is a desirable property for an objective function. However, this particular function becomes undefined if the estimated load on a single link matches the capacity, making it difficult to explore a large space of possible solutions in heavily-loaded networks.

To overcome this problem, we consider a piece-wise linear approximation of $\ell_a/(c_a - \ell_a)$, defined with a heavy penalty for overloaded links. More precisely, for some fixed capacity c_a, we define $\phi(x, c_a)$ as the continuous function with $\phi(0, c_a) = 0$ and derivative in the load ℓ_a of

$$\phi'(\ell_a, c_a) = \begin{cases} 1 & \text{for} \quad 0 \le x/c_a < 1/3, \\ 3 & \text{for} \quad 1/3 \le x/c_a < 2/3, \\ 10 & \text{for} \quad 2/3 \le x/c_a < 9/10, \\ 70 & \text{for} \quad 9/10 \le x/c_a < 1, \\ 500 & \text{for} \quad 1 \le x/c_a < 11/10, \\ 5000 & \text{for} \quad 11/10 \le x/c_a < \infty. \end{cases} \tag{1}$$

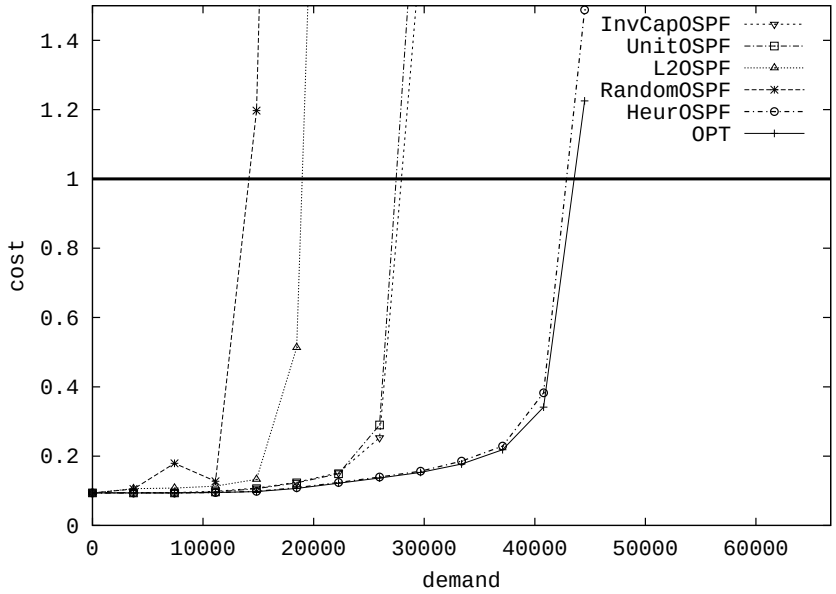


Figure 6: Objective function vs. demand for a proposed AT&T backbone

The link cost function ϕ is illustrated in Figure 5. Generally it is cheap to send flow over a link a with a small utilization ℓ_a/c_a. The cost increases progressively as the utilization approaches 100%, and explodes when we go above 110%. Because of the explosive increase in cost as load exceed capacity, the objective function typically implies that we keep the max-utilization below 1, if at all possible. If we want utilization below 90% rather than 100%, we can simply replace c_a with $c'_a = 0.9c_a$ in (1).

3.2 Fixed Topology and Traffic Demands

Routing based on link weights is not flexible enough to represent all possible solutions to the routing problem. It is relatively easy to construct networks where no setting of the IGP weights can distribute the traffic in an efficient manner [5, 23]. However, these "worst case" examples are quite contrived. The more interesting practical question is how well routing based on static weights performs for realistic topologies and traffic demands. Here, when judging the quality of a weight setting, we compare it against optimal routing (OPT) that can direct traffic along any paths in any proportions. OPT models an idealized routing scheme that can establish one or more explicit paths between every pair of nodes, and distribute arbitrary amounts of the traffic on each of the paths. For our objective function with piecewise-linear link costs, this corresponds to finding the optimal solution to the multi-commodity flow problem. Realizing an OPT solution in practice would require not only a more flexible routing protocol (such as the explicit routes possible with MPLS) but also, in some cases, a rather large number of paths between individual pairs of routers.

Several studies have evaluated algorithms that compute good weight settings for a fixed network topology and a given set of traffic demands [5, 21, 22]. All of these studies found good IGP weight settings that perform within a few percent of OPT in terms of an objective function, though [5] is the only study that evaluates networks with more than 30 routers. Figure 6 shows some of these results for a proposed AT&T backbone with a projected traffic matrix based on traffic measurements and growth trends. Each element in the traffic matrix represents the expected traffic from one router to another. The experiment varied the traffic by multiplying each element by a scaling factor, plotted on the x-axis. The graph plots the value of the objective function normalized to make 1 the threshold for an overloaded network. Each curve corresponds to a different approach to setting the IGP weights:

- **InvCapOSPF:** Each link's weight is inversely proportional to its capacity.
- **UnitOSPF:** Each link's weight is 1.
- **L2OSPF:** Link weight is directly proportional to the physical distance between the incident routers.
- **RandomOSPF:** Each link's weight is random.
- **HeurOSPF:** Link weights are globally optimized with respect to (1).
- **OPT:** The routing is optimal with respect to (1) without requiring the use of shortest paths.

The first three algorithms are reasonable, simple heuristics that do not take the traffic matrix into account. InvCapOSPF attempts to draw traffic toward high-bandwidth links and away from low-bandwidth links. UnitOSPF minimizes the total bandwidth consumption by minimizing the hop count. L2OSPF minimizes the propagation delay for traffic between each pair of routers. RandomOSPF provides a basis for comparison using a naive weight setting. The graph shows that HeurOSPF can handle about 50% higher demands than any of InvCapOSPF, UnitOSPF, L2OSPF, and RandomOSPF before it hits the congestion threshold. OPT performs only slightly better than HeurOSPF, handling about 2% higher demands.

Figure 7 plots the max-utilization metric for the same topology and traffic matrix, using weights optimized based on the objective function in (1). This graph illustrates how the optimization in Figure 6 performs in terms of the most heavily-loaded link in the network. The curve for HeurOSPF closely follows the changes in the piecewise-linear function in (1). The curve remains very close to the curve for OPT before the max-utilization passes 1. The HeurOSPF scheme nicely tries to avoid the high penalty from (1) of 500 for utilizations above 1. On the other hand, HeurOSPF does not need to react when utilization on the link remains below $1/3$. An important point about Figure 7 is that the weight settings found by HeurOSPF worked simultaneously well both for (1) and for max-utilization. In general, we have found that good weight settings are not very sensitive to the exact details of the objective function. Good weight settings according

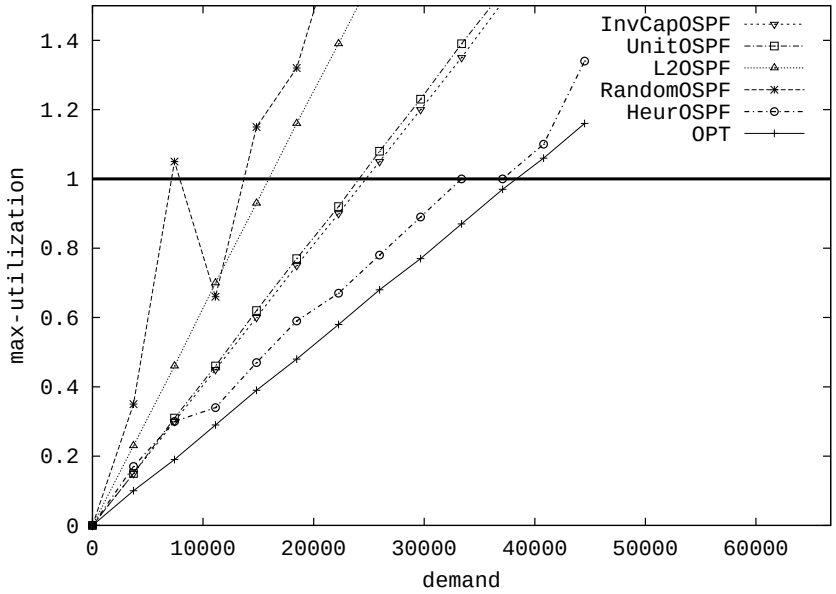


Figure 7: Max-utilization vs. demand for routing optimized based on (1)

to one objective function were simultaneously good for other objective functions, as long as the objective function assigns an increasing penalty to links with load approaching capacity.

3.3 Changing Traffic Demands

Optimizing the weights for a single topology and traffic matrix is not sufficient. In practice, the traffic volumes fluctuate over time and unexpected failures can result in changes to the network topology. In addition, acquiring an exact estimate of the traffic matrix is difficult. It is important that the setting of the IGP weights be robust to changes in the traffic and the topology. To test robustness, we evaluated the optimized setting of the link weights with different traffic matrices. Traffic fluctuations and errors in measuring the traffic matrix were captured by introducing noise. Each element of the traffic matrix was multiplied by a random number between 0 and 2. Although the demands do not change in expectation, this changes each element by 50% on the average. Still, weight settings based on the original traffic matrix continued to perform well for the new input. An optimized setting of the link weights makes good use of the capacity between various parts of the network, and this is not sensitive to small or moderate changes in the traffic. Large shifts in the traffic would require reoptimizing the link weights based on the new traffic matrix.

In addition to statistical fluctuations, the traffic matrix changes throughout the day. For example, example, there may be substantial structural differences between day and evening traffic. We found that IGP

weights optimized for the day-time traffic do not necessarily perform well for the night-time traffic. Instead, we can select a single weight setting that accounts for two different traffic matrices. For example, the optimization can minimize the max-utilization over all links across *both* traffic matrices. In our experiments, we found that a single weight setting could perform quite well for both traffic matrices—almost as well as weights optimized for each matrix individually. Using this single weight setting for the entire day has two main advantages. First, operators do not need to disrupt the network with changes to any of the link weights. Second, the common weight setting performs well for all convex combinations of the two demand matrices and, as such, is effective at accommodating the gradual transitions between day and evening traffic. The approach of optimizing for multiple demand matrices can also be used to select IGP weights that satisfy the requirements of multiple classes of traffic in a network that supports differentiated services, as discussed in [19].

3.4 Small Changes to the Link Weights

Ultimately, changes to the link weights are necessary in response to large shifts in the traffic and certain router or link failures. Limiting the number of weight changes is important to limit the disruption to the network. Minimizing the number of changes is especially important if a human operator needs to evaluate the new weights subject to constraints that are not captured by the objective function. Fortunately, changing a single link weight is often quite effective. When evaluated for an operational AT&T topology, we found that increasing a single weight from 1024 to 1025 could reduce the max-utilization by 8% by diverting some of the traffic to different paths. We also evaluated the effects of all possible link failures to see how often a change in the topology would require a new setting of the weights. In almost every case, the existing IGP weights continued to perform relatively well even after a single link failure. The value of the objective function remained close to the minimum value. However, the failure of a few critical links would require changes to the link weights. A single weight change is often sufficient to alleviate congestion that would arise after a link failure; allowing up to three weight changes was enough to return within a few percent of routing optimized to the new topology. As a proactive measure, the necessary weight changes could be computed in advance of any link failure.

For simplicity, it is appealing to have a small number of different weight values. The InvCapOSPF and UnitOSPF have the advantage of being intuitive and simple. A human operator inspecting the configuration of the router could easily "eye-ball" the setting of the weights and identify any unusual patterns. In addition, having a limited set of values significantly reduces the overhead of an optimization algorithm (such as the one described in the Appendix) that explores possible changes to the weights. In our experiments, we found that having integer values from 1–20 is sufficient to achieve performance that is competitive with OPT. In

fact, a network could operate with a hybrid of the InvCapOSPF and HeurOSPF approach to setting the link weights. Starting with an InvCapOSPF setting of the weights (the default configuration in Cisco routers), we found that changing the weights for ten links was sufficient to perform almost as well as HeurOSPF. This has two important implications. First, operators could still "eye-ball" the configuration, while noting that a few unusual links may have a different weight setting. Second, an optimization algorithm that searches for good weight settings can use the InvCapOSPF configuration as a starting point.

4 Conclusions

Intradomain routing protocols such as OSPF and IS-IS have been deployed in a large number of networks throughout the Internet for many years. In this paper, we have described an approach to engineering the flow of traffic in these networks by monitoring the traffic and topology, optimizing the setting of the static link weights, and reconfiguring the routers with new weight settings as needed. This approach treats traffic engineering as a network operations task, rather than the responsibility of the underlying routing protocol. Working with traditional IGPs has many practical advantages and several experimental studies have shown that they perform well relative to more flexible routing schemes. The approach described in this paper can be applied today in a wide variety of operational networks, and it can help guide the decisions of how to instrument these networks to collect accurate information about the topology and the traffic demands.

The ultimate decision of whether to deploy and use more advanced routing protocols that adapt automatically to the prevailing traffic may depend on a variety of factors that are beyond the scope of this paper. For example, more advanced routing protocols can support route "pinning," which allows the movement of some traffic from one path to another without disrupting the paths for other traffic. In addition, some protocols support the establishment of backup paths, allowing faster re-routing in the event of a network failure. These two enhancements have the potential to reduce the temporary disruption of existing traffic in the face of routing changes and network failures. The features may be useful for networks that support real-time applications or have relatively frequent equipment failures. The performance benefits and operational complexity of these enhancements need to be better understood. The comparison between HeurOSPF and OPT can help inform the discussion of the cost-performance trade-offs. In the end, some hybrid may emerge where schemes like HeurOSPF are used to select the configuration of the base link weights and dynamic protocols are used to provide additional information about the current load on each link.

Traffic engineering with traditional IGPs has a number of interesting avenues for future research work. Having a single, integer weight on each link is surprisingly effective in controlling the flow of traffic in realistic networks. New theoretical work could potentially provide a more formal explanation for why this is true for certain classes of graphs. In addition, it would be useful to understand how far this simple

paradigm can be pushed to incorporate additional constraints, such as propagation delay. Other interesting research areas include the convergence behavior of OSPF and IS-IS. This has been considered in some initial work in [17, 18], though our understanding of IGP convergence delay in operational networks is still quite limited. The design and deployment of new techniques that minimize the delay in detecting failed links would be quite valuable, both for our approach and for schemes based on more advanced routing protocols. In addition, new work could identify techniques that minimize the influence of routing changes on the forwarding of existing traffic. Progress in these areas of research would be extremely valuable in helping operators run their networks more efficiently.

A Optimization Algorithm

In this appendix, we briefly describe the algorithm used to optimize the link weights for the HeurOSPF scheme in Section 3. First consider the simple setting from Section 3.2 of a fixed topology and demand matrix. Finding an optimal setting of the link weights is computationally intractable, in the sense that approximating the optimal solution is NP-hard. As a result, we cannot expect to find an algorithm that is guaranteed to be both fast and produce close to optimal weight-settings. Instead, in [5], we resorted to a local search heuristic [24]. Local search heuristics do not come with any guarantees, neither for speed nor for quality, yet they work well in practice for many combinatorial optimization problems.

The basic principle of local search is to iteratively generate and evaluate candidate solutions. In each iteration, they define a neighborhood of candidate solutions each of which is a small modification of the current solution. The next candidate solution is chosen from the neighborhood. In our case, a candidate solution is IGP configurations, i.e., an assignment of link weights. Each neighbor is obtained by changing one, or a few, link weights.

It is desirable that the next solution improves the objective function. Among techniques using only moves that reduce the objective function, descent methods consider the entire neighborhood, select an improving neighbor (usually the best one), and stop when a local minimum is found. However, this local minimum can be far away from the globally optimal configuration. Several techniques have been proposed that allow non-improving moves to avoid becoming trapped in a local minimum. Local search heuristics such as tabu search and simulated annealing allow such non-improving moves while applying restrictions to the neighborhood to prevent cycling. We allowed non-improving moves, using hashing to avoid cycling.

The bottleneck in finding a good IGP configuration with a local search heuristic is the need, for each candidate set of link weights, to compute the resulting paths, estimate the resulting flow of traffic, and evaluate the objective function. However, when evaluating a neighbor of a current candidate weight setting, we can benefit from the fact that the neighbor only differs in a few weights. This is exploited by incremental

algorithms that instead of evaluating the neighbor from scratch, reuses the current evaluation and only re-evaluates the parts affected by weight changes.

As discovered recently [25, 26], there are several other meta-heuristics that can be used to find good IGP weight settings.

In [19], we use the same local search to find weight settings that are good with respect to multiple demand matrices (c.f. 3.3). Also, we modify the local search to improve the value of the objective function with as few weight changes as possible (c.f. 3.4). From a programming perspective, a main advantage of the local search from [5] is that it could easily be adapted to the different configurations in [19]. The optimization algorithm presented in these two papers has been incorporated in the NetScope tool developed at AT&T [16].

Acknowledgments

We would like to thank Randy Bush for his feedback on an early outline of this paper, and Aman Shaikh for his comments on our initial draft. We are also grateful to Jay Borkenhagen from AT&T for his help in understanding the operational constraints for configuring OSPF in large IP networks.

References

[1] D. O. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, "Overview and principles of Internet traffic engineering." Request for Comments 3272, May 2002.

[2] D. O. Awduche, "MPLS and traffic engineering in IP networks," *IEEE Communication Magazine*, pp. 42–47, December 1999.

[3] D. Katz, D. Yeung, and K. Kompella, "Traffic engineering extensions to OSPF." Work in progress, Internet Draft draft-katz-yeung-ospf-traffic-06.txt, Expires April 2002.

[4] T. Li and H. Smit, "IS-IS extensions for traffic engineering." Work in progress, Internet Draft draft-ietf-isis-traffic-04.txt, August 2001.

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

[6] A. Feldmann and J. Rexford, "IP network configuration for intradomain traffic engineering," *IEEE Network Magazine*, pp. 46–57, September/October 2001.

[7] A. Shaikh, M. Goyal, A. Greenberg, R. Rajan, and K. K. Ramakrishnan, "An OSPF topology server: Design and evaluation." To appear in *IEEE Journal of Selected Areas in Communications*, special issue on Recent Advances on Fundamentals of Network Management, 2002.

[8] M. Grossglauser and J. Rexford, "Passive traffic measurement for IP operations." To appear in *The Internet as a Large-Scale Complex System*, `http://www.research.att.com/~jrex/papers/sfi.ps`, 2002.

[9] X. Xiao, A. Hannan, B. Bailey, and L. Ni, "Traffic engineering with MPLS in the Internet," *IEEE Network Magazine*, pp. 28–33, March 2000.

[10] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True, "Deriving traffic demands for operational IP networks: Methodology and experience," *IEEE/ACM Trans. Networking*, vol. 9, June 2001.

[11] N. Taft, S. Bhattacharyya, J. Jetcheva, and C. Diot, "Understanding traffic dynamics at a backbone POP," in *Proc. Workshop on Scalability and Traffic Control in IP Networks, SPIE ITCOM+OPTICOMM Conference*, August 2001.

[12] J. Cao, D. Davis, S. V. Wiel, and B. Yu, "Time-Varying Network Tomography," *Journal of the American Statistical Association*, December 2000.

[13] N. G. Duffield and M. Grossglauser, "Trajectory sampling for direct traffic observation," *IEEE/ACM Trans. Networking*, vol. 9, pp. 280–292, June 2001.

[14] J. W. Stewart, *BGP4: Inter-Domain Routing in the Internet*. Addison-Wesley, 1999.

[15] S. Halabi and D. McPherson, *Internet Routing Architectures*. Cisco Press, second ed., 2001.

[16] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, and J. Rexford, "NetScope: Traffic engineering for IP networks," *IEEE Network Magazine*, pp. 11–19, March 2000.

[17] J. Parker, D. McPherson, and C. Alaettinoglu, "Short adjacency hold times in IS-IS." Expired Internet Draft draft-parker-short-isis-hold-times-01.txt, July 2001.

[18] C. Alaettinoglu, V. Jacobson, and H. Yu, "Towards milli-second IGP convergence." Expired Internet Draft, draft-alaettinoglu-isis-convergence-00.txt. `http://www.packetdesign.com/Documents/convergence.pdf`.

[19] B. Fortz and M. Thorup, "Optimizing OSPF/IS-IS weights in a changing world," *Journal of Selected Areas in Communications (Special Issue on Recent Advances on Fundamentals of Network Management)*, vol. 20, no. 4, pp. 756–767, 2002.

[20] B. Fortz and M. Thorup, "Increasing Internet capacity using local search," Tech. Rep. IS-MG 2000/21, Université Libre de Bruxelles, 2000. `http://smg.ulb.ac.be/Preprints/Fortz00_21.html`.

[21] F. Lin and J. Wang, "Minimax open shortest path first routing algorithms in networks supporing the SMDS services," in *Proc. International Conference on Communications*, vol. 2, pp. 666–670, 1993.

[22] K. Ramakrishnan and M. A. Rodrigues, "Optimal routing in shortest-path data networks," *Bell Labs Technical Journal*, vol. 6, January–June 2001.

[23] D. H. Lorenz, A. Orda, D. Raz, and Y. Shavitt, "How good can IP routing be?," Tech. Rep. 2001-17, DIMACS, May 2001.

[24] E. Aarts and J. Lenstra, *Local Search in Combinatorial Optimization*. Wiley-Interscience, 1997.

[25] M. Ericsson, M. Resende, and P. Pardalos, "A genetic algorithm for the weight setting problem in OSPF routing," *J. Combinatorial Optimization*, vol. 6, no. 3, pp. 299–333, 2002.

[26] L. S. Buriol, M. G. C. Resende, C. C. Ribeiro, and M. Thorup, "A memetic algorithm for OSPF routing," in *Proceedings of the 6th INFORMS Telecom*, pp. 187–188, 2002.