Cautious Weight Tuning for Link State Routing Protocols

SICS Technical Report T2011:01
ISSN 1100-3154

Anders Gunnar
Swedish Institute of Computer Science
P.O. Box 1263, SE-164 29 Kista, Sweden
Email: aeg@sics.se

Mikael Johansson
School of Electrical Engineering, KTH
SE-100 44, Stockholm, Sweden
Email: mikaelj@ee.kth.se

January 26, 2011

Abstract

Link state routing protocols are widely used for intradomain routing in the Internet. These protocols are simple to administer and automatically update paths between sources and destinations when the topology changes. However, finding link weights that optimize network performance for a given traffic scenario is computationally hard. The situation is even more complex when the traffic is uncertain or time-varying. We present an efficient heuristic for finding link settings that give uniformly good performance also under large changes in the traffic. The heuristic combines efficient search techniques with a novel objective function. The objective function combines network performance with a cost of deviating from desirable features of robust link weight settings. Furthermore, we discuss why link weight optimization is insensitive to errors in estimated traffic data from link load measurements. We assess performance of our method using traffic data from an operational IP backbone.

1 Introduction

Link state routing protocols are transparent, simple to administer and possess a remarkable ability to recover from component failures. After a failure, routers

autonomously find a new consistent routing in a fully distributed manner. However, link state routing protocols also have disadvantages. The shortest path principle that enables each router to find a consistent routing limits the routes that can be realized using link state routing. Furthermore, it is computationally hard to find link weights that give optimal network performance even when traffic patterns are known.

In this paper we study search heuristics for parameter setting in two of the most widely used routing protocols in the Internet today, Open Shortest Path First (OSPF) and Intermediate System Intermediate System (IS-IS). Both these protocols are link state routing protocols where the network is modeled as a graph whose nodes represent routers and edges represent links connecting the routers. Each link is assigned a weight reflecting the cost of sending traffic over the link and routes are computed such that traffic flows along the shortest path from source to destination. These shortest paths are typically computed using Dijkstra's algorithm. To tune the link weights for improved network performance one needs to understand how traffic flows in the network. However, constant monitoring of traffic in a large IP backbone can be challenging and resource consuming. Hence, many operators only perform occasional measurements on selected parts of the network. These partial and occasional measurements, combined with the time varying nature of Internet traffic lead to an uncertainty of the traffic situation. In addition, the interplay between internal routing within a network and the routing between administrative domains is known to cause large shifts in traffic patterns [15]. Thus, it is desirable to find parameter settings that reduce performance degradations for a wide range of load variations. Contrary to *e.g.* [6] we do not optimize for the normal traffic situation but try to minimize the worst network performance that can occur under any foreseeable traffic scenario. This is novel also with respect to other relevant work: Balon and Leduc [4] address traffic uncertainty caused by the interplay between intra and interdomain routing but do not consider the worst case traffic scenario; previous work on robust routing such as COPE [17], oblivious routing [3] and the authors' previous work [7, 8], do not apply to link state routing protocols.

With our previous experience from robust routing without the constraints of link state routing ([7, 8]) it is natural to try to develop similar algorithms for optimizing weights in link state routing protocols. To evaluate our approach we focus on uncertainties arising from interdomain reroutes. We know from our earlier work [7] that this type of uncertainty can be dealt with efficiently when the routing does not need to follow shortest paths. In this paper, we leverage on our previous work to develop efficient search heuristics for finding weight settings with guaranteed performance for all foreseeable traffic scenarios. We call our approach *cautious weight tuning* since in each step of the algorithm the weight change that gives the largest guaranteed performance improvement for every possible traffic scenario is executed. Furthermore, we explain why optimization of link weights is robust to estimation errors in estimated traffic matrices from link load measurements.

The rest of the paper is organized as follows. In the next section we define

cautious weight tuning and the notation used in the paper. Section 3 describes our approach. A use case with interdomain reroutes and a numerical example from an operational IP backbone is presented in Section 4. Cautious weight tuning under traffic matrix uncertainty is studied in Section 5 and related work is discussed in Section 6. Finally, we conclude our findings in Section 7.

2 Notation and problem formulation

The network is represented by a graph (N, E) where N is the set of nodes and E is the set of edges representing routers and links, respectively. The capacity of a link $l \in E$ is denoted c_l. Furthermore, the traffic demand between source-destination pair p is denoted s_p and the set of source destination pairs P has $|N|(|N|-1)$ elements. In addition, the traffic demands are assumed not to be known with complete certainty. Instead, the traffic demands belong to an uncertainty set $\mathcal{S}$.

We assume that routing is performed by a link state routing protocol, such as OSPF or IS-IS. Associated to each link l is a weight w_l that describes the cost of sending data across that link. Data is routed over the shortest paths in this link metric. The routes for a given weight setting can be represented by an $|E| \times |P|$ routing matrix $R = [r_{lp}]$ where each element r_{lp} represents the fraction of traffic demand s_p routed over link l. With this notation, the total traffic t_l across link l is

$$t_l = \sum_{p \in P} r_{lp} s_p = r_l^T s$$

where r_l^T is the lth row of the routing matrix R. Thus the vector $t = [t_l] \in \mathbb{R}^{|E|}$ of total traffic is given by the equation

$$t = Rs. \tag{1}$$

When link state routing with a single path between source and destination is used, the elements in R assume values 0 or 1 depending on whether a source-destination pair is routed on the link or not. Adjusting the link weights changes the routing matrix R and alters the total traffic t across links.

Since the traffic s is uncertain, it is hard to predict the traffic load that results from a specific change in the link weights (and hence in the routing matrix). In this work, we consider *cautious weight tuning*, where we attempt to find link weights that are guaranteed to improve system performance for all traffic scenarios in the uncertainty set. In each step of the tuning, one link weight at a time is adjusted to form a new weight setting w^+ and the corresponding routing matrix R^+ is calculated. For this routing we determine the worst case traffic scenario $s_{wc} \in \mathcal{S}$. The weight setting that gives the largest performance improvement is executed and the procedure is repeated. Cautious weight tuning was first introduced in [9] for weight tuning under traffic matrix uncertainty, but the algorithm proposed in that paper often fails to find weight changes that

guarantee improved performance. In this paper we generalize the applicability of cautious weight tuning and enhance the search with a novel objective function.

3 Cautious weight tuning

Even under the assumption that the traffic matrix is known, link weight optimization is computationally hard. One can either approach the problem formally via mixed-integer-linear programming models [11] or using search heuristics [5]. Since the computational burden of the integer programming approach quickly becomes prohibitive as network size grows, search heuristics are often preferred in practice. Search heuristics do not certify optimality for the obtained solution but experiments indicate that it is possible to find near-optimal solutions with a reasonable computational effort [1, 5].

We base our evaluation on a search technique first introduced by Fortz and Thorup [5], referred to as FT in the rest of the paper. FT is a local search algorithm where neighboring weight settings are created by changing a single weight and the new weight settings are evaluated using their predicted network performance. Our method has two distinct differences: first, we use the worst-case traffic over the full uncertainty set as the performance measure; second, we do not evaluate routing settings based on their predicted network performance only, but also account for properties of the routing that we know are more likely to give robust solutions.

A key step in our approach is to determine the worst-case traffic scenario for a given weight setting. Using link utilization ($u_l = t_l/c_l$) as performance metric the worst case traffic scenario can be found by solving, for each link $l \in E$, the optimization problem

$$\begin{array}{ll} \text{maximize} & c_l^{-1}(r_l^+)^T s_{wc} \\ \text{subject to} & s_{wc} \in \mathcal{S} \\ & s_{wc} \succeq 0 \end{array} \tag{2}$$

where $(r_l^+)^T$ is the l^{th} row in the routing matrix for weight setting w^+. If $\mathcal{S}$ is a convex set then s_{wc} can be found in polynomial time using modern interior-point methods. In particular, if $\mathcal{S}$ can be described as the solution set of a set of linear equations (2) becomes a linear programming problem.

The straightforward robust version of FT would work as follows. In each iteration, neighboring weight settings are computed and their worst-case performance are computed by solving (2). The weight setting that guarantees the lowest link utilization over all traffic scenarios is executed, and the procedure is repeated until the stopping criterion is fulfilled.

A drawback with this method is that it is hard to find single weight changes that actually improve worst-case performance. In other words, multiple weight settings might be needed before the worst-case performance is improved. Including multiple weight changes in the search algorithm drastically increases

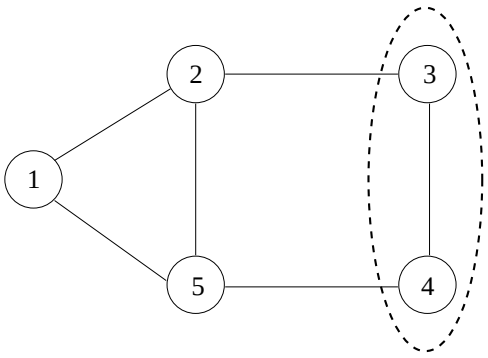

Figure 1: Simple example network with five nodes, one group of possible egress nodes for traffic inserted in the network at node one and two

the search space, and hence the computational requirements on the algorithm. The key idea of our method is to stay with single weight changes in each search step but include an additional term in the search objective that discourages weight changes that are likely to be sensitive to traffic uncertainty. For example, in many cases we know that we should discourage weight changes that split up traffic of known volume into subflows of unknown volume. The following example illustrates the idea.

We consider the simple five node example network depicted in Figure 1. Ten units of traffic is injected in node one and two, however, the traffic can be destined to either node three or node four. This type of uncertainty appears for interdomain routing where routes to a destination outside the network domain are available in several locations in the network. Depending on how the interdomain routing protocol selects preferred route, traffic to a destination may leave the network in several locations. The uncertainty of the traffic demands can be expressed by the equations

$$\begin{aligned} s_{13} + s_{14} &= 10 \\ s_{23} + s_{24} &= 10. \end{aligned}$$

If we set the weight on each link to one both traffic demands s_{13} and s_{23} are routed on the link between node two and three. This will result in a worst case link load of 20 units of traffic. However, with careful tuning of link weights it is possible to let traffic demand s_{14} follow the path 1→5→4 and demand s_{13} path 1→5→4→3. Assuming demands s_{23} and s_{24} take the paths 2→3 and 2→3→4 respectively we are able to reduce the worst case link load to ten units of traffic. Thus, from the example we conclude that it is desirable to route flows s_{13} and s_{14} together while demands s_{23} and s_{24} are routed together on separate links from the other two traffic demands. The challenge is to obtain a weight setting that is able to realize such a routing.

Based on our observations in the example above we add a penalty function to the search objective which encourages known traffic flows to be routed to-

gether. To this end, let R be a routing matrix calculated from a given weight setting using the shortest path principle. The $|P| \times |P|$ matrix

$$Q = R^T R - \text{diag}(R^T R) \tag{3}$$

will then, for each matrix element $[Q]_{ij}$ where $i \neq j$, describe the number of links shared by the shortest path routes for traffic demands s_i and s_j. The diagonal elements of the matrix $R^T R$ are the lengths of the individual paths. Since we do not penalize the lengths of individual paths (nor encourage long paths) in this paper, we make sure that the diagonal elements of Q are zero. We also define a matrix $C \in \mathbb{R}^{|P| \times |P|}$ that encodes which traffic flows we encourage to share routes and which traffic flows we discourage from sharing routes. Specifically, the elements of C are given by

$$[C]_{ij} \begin{cases} = 1 & \text{if demand } i\text{and } j \text{ should be routed together} \\ = -1 & \text{if demand } i\text{and } j \text{ should } \textbf{not} \text{ be routed together} \\ = 0 & \text{otherwise.} \end{cases}$$

In general, determining what traffic demands to route together have to be tailored to the traffic uncertainty at hand. This requires insight or intuition about the specific problem. We will show shortly how the elements of C can be set to generate routing settings that are robust to traffic shifts due to intradomain reroutes. Finally, we define a hint function

$$h(Q, C) = \text{Tr}(CQ) \tag{4}$$

where $\text{Tr}(\cdot)$ is the trace operator. The function $h(Q, C)$ serves as a hint to help the heuristic favor routing settings that route certain flows together to decrease uncertainty of the load on the link. The hint function gives a high value if the routing determined by R routes a high number of desired flows together and penalizes routing settings where a lower number of flows are routed together.

The objective function used by our search algorithm is thus

$$\text{obj}(w) = u_{\max}(w; \mathcal{S}) - \kappa h(Q(w); C). \tag{5}$$

Here, $u_{\max}(w; \mathcal{S})$ is the maximum worst-case link utilization for shortest path routing with link weights w (determined by solving the optimization problem (2) for every link in the network) while h is the hint function defined above. The parameter κ determines how much emphasis should be given to the hint in comparison to the link utilization. If κ is set to zero, we disregard influence of the hint function.

For each iteration of the search heuristic $|E|$ different weight settings are tested, and the weight setting that gives the largest decrease in the search objective is executed. The actions taken for a weight setting can be summarized as follows:

1. Produce a neighboring weight setting w^+ and calculate the corresponding routing matrix R^+.

2. Determine the worst case traffic scenario s_{wc} for R^+.
3. Evaluate the objective function for R^+ and s_{wc}.

The iterations are repeated until a stopping criterion is satisfied. Note that a property of our objective function is that it accepts weight settings that give an increase in link utilization if the hint function is improved sufficiently much. This also means that the weight setting produced during the last iteration might not necessarily be the best (since the hint function might have caused a change). To be able to recall the best weight setting we keep track of the best $u_{\max}$ and the associated weight setting during our search.

4 Application of cautious weight tuning: weight setting under BGP traffic uncertainty

For resilience, network operators often connect with other operators in several different locations to exchange traffic and routing information. The routing information is encoded as prefixes representing reachable destination networks. As a result of introducing multiple connection points, many prefixes are announced (and hence available for forwarding traffic) in several locations in the network. Depending on the setting of interdomain routing protocol attributes and configuration of intradomain routing protocol parameters traffic take a selected route for each prefix. Since conditions may change due to *e.g.* reconfigurations, failures or withdrawn routes, there is uncertainty about how the traffic will flow in the network. It has been shown in many studies, that interdomain reroutes may cause large traffic shifts in the network (*e.g.* [14, 15]). These uncertainties can efficiently be handled for routing without the constraints from shortest path routing [7]. In this section we investigate if this also holds for link state routing protocols.

We assume that at every ingress router it is possible to measure the amount of traffic destined to each destination prefix in the routing table using flow measurement functionality such as Cisco's Netflow. Since flows of known volume should be routed together, traffic from a common source destined to a network prefix announced by multiple egress routers is routed together. A difficulty is that a routing table in the default free zone in the Internet today contains in the order of 200 000 prefixes. Creating a model where all prefixes are included would create an intractable model. To reduce problem size we could use the methodology described in [7] where it is observed that most of the prefixes in the routing table route negligible amounts of traffic. However, we address the scalability problem in a different fashion by observing that prefixes are typically announced by a limited number of groups of peering points. In Figure 2 there are three groups of peering routers announcing three prefixes. In this example we replace three prefixes with three groups of egress routers for traffic to the prefixes. However, if a group contains a large number of prefixes it is possible to obtain a large reduction of parameters to describe the traffic uncertainty

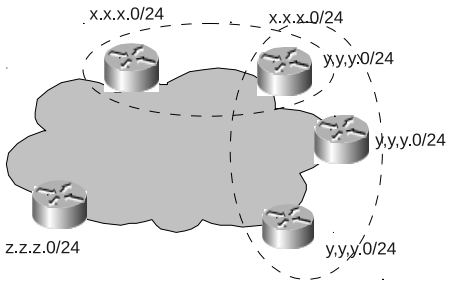

Figure 2: Example network with three groups of egress points for prefixes x.x.x.0/24, y.y.y.0/24 and z.z.z.0/24 respectively

by aggregating traffic from a source destined to prefixes in each group.

To formulate the model in equations we let t_{og} denote the amount of traffic sent from ingress router o to egress router group $g \in \mathcal{G}$ where $\mathcal{G}$ is the set of groups of peering routers announcing prefixes. Furthermore, $e(g)$ is the set of egress routers of group g. The variable δ_{odg} represents the fraction of traffic from source o to group g that leave the network at egress router d assuming $d \in e(g)$. If $d \notin e(g)$ then $\delta_{odg} = 0$. To identify the worst case traffic matrix for a given routing matrix R we solve the following optimization problem for each link l in the network

$$\begin{aligned} \text{maximize} \quad & c_l^{-1} r_l^T s \\ \text{subject to} \quad & \sum_{g \in \mathcal{G}} t_{og} \delta_{odg} = s_{od}, \forall o, d, o \neq d \in N \\ & \sum_{d \in e(g)} \delta_{odg} = 1, \forall g \in \mathcal{G}, \forall o \in N \\ & \delta_{odg}, s_{od} \geq 0 \end{aligned} \tag{6}$$

where δ_{odg} and s_{od} are the optimization variables and the constraints in the optimization problem constitute the traffic uncertainty set $\mathcal{S}$. The worst case is the traffic scenario that gives the highest link utilization for all $l \in E$. In our analysis we found that for the 163 000 prefixes in our routing data set we were able to identify 35 groups of routers announcing prefixes. Thus, we are able to reduce the number of variables considerably in the optimization problem.

Based on the observations made in Section 3 we conclude that source destination traffic demands s_{od} from a common source o and destined to a destination router d that belong to a common group $g \in \mathcal{G}$, should be routed together. To accomplish this we proceed as follows:

1. For each source router group together routers announcing the same pre-

fixes, and set their corresponding elements in the C-matrix to 1.

2. Traffic demands destined to a router in an egress group but from different sources have their corresponding elements in C set to -1.

3. Other elements in C are set to zero.

Although this is a rather simplistic approach we will see later that it will serve our purposes well. Also note that if groups are partially overlapping, the values of involved elements in C will not be uniquely determined. However, we neglect such considerations here.

4.1 Numerical examples

For the evaluation we use data from the GEANT network provided by Uhlig *et al.* [16]. The GEANT network connects national research networks in Europe and has 23 nodes and 74 links. In our experiments we have access to network topology, traffic data and BGP routing information. The traffic measurements were performed during a four month period, and consist of 15 minute flow exports of sampled Netflow measurements where one out of every thousand packets is sampled. Furthermore, the BGP routing information base is recorded every day during the measurement period. Since we use maximum link utilization as objective in our optimization, we upgrade links of 155 Mbps to 2400 Mbps as these links would otherwise remain bottlenecks irrespectively of the routing. More details about the data set can be found in [16].

4.1.1 Comparison with other approaches

We compare cautious weight tuning with weight tuning for a nominal traffic scenario obtained from the original link weights used by BGP to select egress point for each prefix in the routing table (i.e. the FT heuristic for the nominal traffic scenario). For comparison we also provide results of optimal robust routing under interdomain reroutes without the constraints imposed by link state routing [7]. To demonstrate the benefit of robust routing, we also provide results where a routing optimized for the nominal case is subjected to the worst-case traffic scenario in $\mathcal{S}$.

Figure 3 displays $u_{\max}$ for optimization for the nominal case (nomOPT), robust optimization (robOPT) [7], weight tuning using the heuristic by Fortz and Thorup (FT) and cautious weight tuning using FT (cautiousFT). The results for nomOPT indicate that although large performance gains can be made from optimization for the nominal case, this routing setting is highly sensitive to deviations from normal operation. The worst case link utilization is almost twice as high as expected for the nominal case it was optimized for. Similar results can be observed for the weight tuning using FT. For robust optimal (robOPT) the worst case link utilization is reduced but utilization under normal operation has increased from 0.24 to 0.35 in this experiment. Furthermore, performance of cautiousFT is almost identical to optimal robust routing.

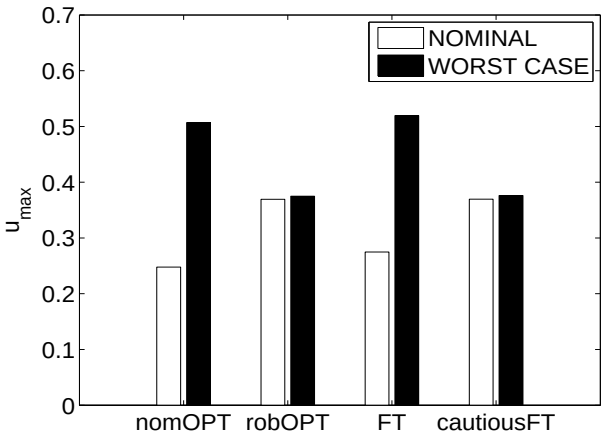

Figure 3: Maximum link utilization for different routing settings and traffic situations, nominal traffic scenario (White) and worst case traffic scenario (Black)

4.1.2 Progress

Figure 4 displays progress of cautiousFT for each iteration of the algorithms and best possible setting of κ. The vertical axis indicate deviation of $u_{\max}$ from optimal obtained by robOPT. The weight tuning is set to terminate after 100 iterations. However, the best weight setting is found after 53 iterations of the algorithm. We note that performance gains are made in steps after a number of iterations have been executed. In order to reduce $u_{\max}$ under traffic uncertainty a number of weight changes need to be executed before a performance gain in link utilization can be observed. During the periods when no progress in $u_{\max}$ is made, the hint function guides the heuristic in a direction where progress can be obtained. In Figure 4 it can be noted that at some instances slightly worse weight settings are accepted by the algorithm. In these situations the hint function outperforms the value of maximum link utilization.

4.1.3 Tuning the parameter κ

A critical component of cautious weight tuning is setting the parameter κ. Figure 5 shows deviation of $u_{\max}$ from optimal value obtained using the algorithm from [7] as a function of κ, and κ is plotted in logarithmic scale. The plot reveals that satisfactory performance is obtained for a wide variety of values of κ. Our findings indicate that weight settings that are robust to traffic shifts due to interdomain reroutes exist and with the right settings of the matrix C and the parameter κ it is possible to find these weight settings using well-established search techniques.

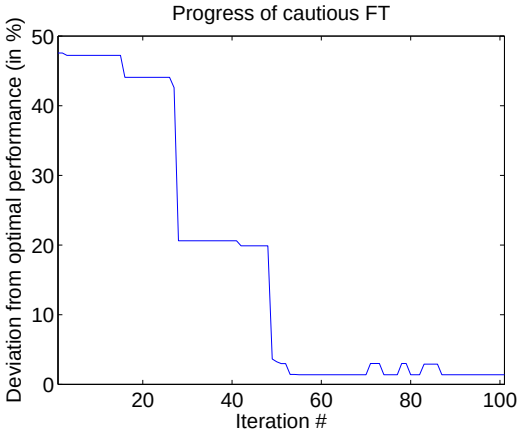

Figure 4: Deviation from optimal performance in $u_{\max}$ for each iteration of cautious FT

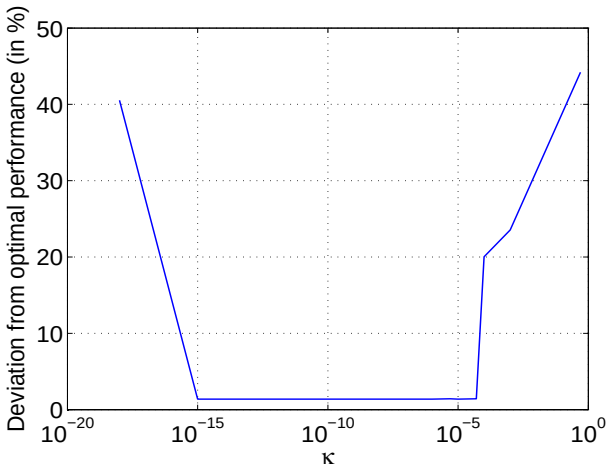

Figure 5: Deviation from optimal performance in $u_{\max}$ as a function of the hint multiplied by the parameter κ for cautiousFT

Table 1: Execution time, fraction of changed weights (from start weights) and number of identified worst case traffic scenarios for cautiousRR and cautiousFT. Parameter κ set to obtain best possible performance

	Exec. time (Sec.)	Changed weights	s_{wc} identified
cautiousFT	1972	35%	113886

4.1.4 Computational considerations

Unfortunately we have not access to detailed traffic data from other networks than the GEANT network. Hence, we are not able to perform a detailed evaluation of the computational burden of our proposal for traffic uncertainty caused by interdomain reroutes. However, Table 1 summarizes some computational aspects of cautiousFT. The fraction of changed weights is the fraction between the number of changed weights and total number of link weights in the network. Furthermore, the total number of variables and equality constraints in problem (6) are 1826 and 1289 respectively. Note that the exact number of variables and constraints depend on the composition of the groups $g \in \mathcal{G}$ of egress routers.

5 Cautious weight tuning under traffic matrix uncertainty

In the light of the encouraging results on robust routing [9] and the results of optimization of OSPF/IS-IS link weights from estimated traffic demands [13], the rather disappointing results of cautious weight tuning in [9] might appear surprisings. To explain these findings we return to the determination of the worst case link traffic scenario as formulated in (2).

To illustrate how changes in link-weights influence the variability in worst case traffic scenarios we randomly select link weights in the GEANT network and set the weight to a random number between 1 and 1000. Assuming R_{est} and t_{est} are known, $u_{\max}$ for the nominal traffic scenario as well as an identified worst case traffic scenario in the solution space of $R_{\text{est}} s_{wc} = t_{\text{est}}$ is identified by solving the optimization problem

$$\begin{array}{ll} \text{maximize} & c_l^{-1}(r_l^+)^T s_{wc} \\ \text{subject to} & R_{\text{est}} s_{wc} = t_{\text{est}} \\ & s_{wc} \succeq 0 \end{array} \tag{7}$$

for each row r_l^+ in the adjusted routing matrix R^+. Link utilization for s_{wc} is denoted $u_{\max}^{wc}$. Figure 6 plots the average value of $u_{\max}^{wc}/u_{\max}$ for 100 different routing matrices for different levels of random changes of link weights. The average value is close to one up to when 50 percent of the link weights are changed. However, there is also a high degree of variability in the results.

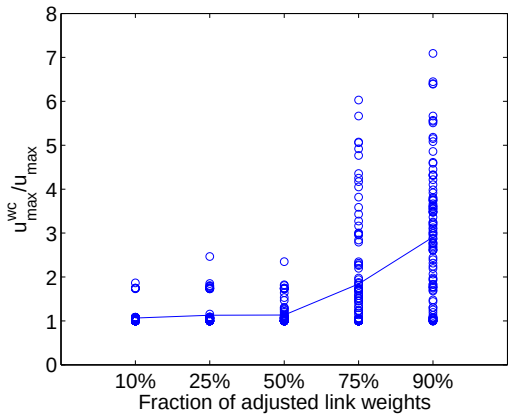

Figure 6: $u^{wc}_{\max}/u_{\max}$ when 10,25,50,75,90% of link weights are changed to a random integer. Average value (solid line) and samples plotted for each level

Nevertheless, the experiment indicate that maximum link utilization is rather insensitive to changes in the link weights.

To explain these observations it is instructive to consider the null space of the original routing matrix R_{est}. The solution set to $R_{\text{est}}s = t_{\text{est}}$ can be described as the sum of a particular solution s_0 satisfying the constraints and a linear combination of the vectors in the null-space of R_{est}. The solution set can be described by the equation

$$s = s_0 + (I - R^{\dagger}_{\text{est}} R_{\text{est}})z \tag{8}$$

where $R^{\dagger}_{\text{est}}$ is the pseudo-inverse of the matrix R_{est} and z are coordinates such that $s \succeq 0$. We replace s with (8) and use the property of the pseudo-inverse: $R_{\text{est}} = R_{\text{est}} R^{\dagger}_{\text{est}} R_{\text{est}}$. After simplifications we arrive at the following optimization problem

$$\begin{array}{ll} \text{maximize} & c_l^{-1}(r_l^+)^T(I - R^{\dagger}_{\text{est}} R_{\text{est}})z \\ \text{subject to} & s_0 + (I - R^{\dagger}_{\text{est}} R_{\text{est}})z \succeq 0 \end{array} \tag{9}$$

The optimization problem (9) only takes the variability due to the uncertainty into consideration. We note that if a row in the new routing matrix is unchanged, the objective will be identical to zero due to properties of the pseudo-inverse mentioned above. This helps to explain why cautious weight tuning is difficult under traffic matrix uncertainty as observed in [9]. A small change in the a link weight only makes a small change in the routing matrix. Thus, the objective function in problem (9) will be identical to zero for most of the links. Furthermore, maximum link utilization as objective function takes the most congested link into consideration only. Changes in other parts of the network have no influence on the objective except when another link becomes the

bottleneck. Thus, the robustness of link state routing protocols to link load measurement traffic uncertainty observed [1, 13] seems to be connected to the choice of maximum link utilization as objective function. However, it is likely that other properties specific to link state routing also play an important role. For instance, since paths sharing links before a weight change will to a large extent continue to share links after a weight change due to the rules of SPF routing.

6 Related work

One of the earliest and also one of the most cited papers on weight setting for link state protocols is Fortz and Thorups paper on local search heuristics [5]. The local search heuristics are extended to find weight settings for a wider selection of traffic situations in [6]. Ramakrishnan and Rodrigues [12] use a descending search algorithm where in each step one flow is deviated from a link in order to decrease a cost function. Another often cited paper is Wang *et al.* [18] where the Lagrangian variables obtained from a dual optimization problem is interpreted as link weights. Abrahamsson and Björkman [2] use a two step cost function which strives to keep load in the network under a prescribed level. In addition, the search heuristic is a combination of the heuristics by Fortz and Thorup [5] and Ramakrishnan and Rodrigues [12]. To handle reroutes caused by hot potato routing, Balon and Leduc [4] design a novel cost function that attempts to compensate for these effects. Nucci *et al.* [10] design a cost function that optimizes not only for the normal network topology but also for a number of different fault scenarios that might occur where links or nodes fail.

A somewhat different approach is taken by Xu *et al.* [19] where DEFT is introduced. With DEFT traffic can be sent over non shortest paths using an exponential penalty function. However, DEFT require minor modifications of the OSPF/IS-IS protocols.

7 Conclusion

In this paper we develop *cautious weight tuning* for link state routing protocols such as the widely used OSPF and IS-IS routing protocols for intradomain routing in the Internet. With cautious weight tuning it is possible to optimize link weights for a set of traffic scenarios to take into account variability and uncertainty in traffic data. Our work differ from previous studies (e.g. [4, 6]) in the sense that we explicitly identify the worst case traffic scenario and optimize the routing for this case. In other words, our approach does not only optimize the network for normal operation, but attempts to find routing settings that guarantee a certain performance for all foreseeable traffic patterns. Such routing settings allow a network operator to provision a more predictable and reliable service even when the traffic changes dramatically. To guide the heuristics we augment the desired network performance objective with a hint function that

captures desirable properties of a robust routing setting. We highlight performance of the augmented objective function using a well-known search heuristic. In addition, we present some evidence of why optimization of link weights is robust to errors in traffic data caused by estimation of the traffic matrix from link load measurements. However, other properties may also influence the robustness of link state routing but this requires more investigation. The robustness to estimated traffic matrices of link state routing has been observed by many researchers before (e.g. [1, 13]) but to the best of our knowledge no explanation has been presented.

Although initial results presented in this paper are promising they should be considered preliminary. Further evaluation is needed on other network topologies and traffic situations to fully assess our approach. Other types of traffic uncertainties should be considered as well as a more general method to determine the elements in the C matrix.

Acknowledgment

This work was supported in parts by the SICS Center for Networked Systems, the Swedish Research Council and the European Commission. The authors would like to thank Dr. Martin Nilsson for valuable suggestions and discussions during the development of this paper.

References

[1] A. Gunnar and H. Abrahamsson and M. Söderqvist. Performance of Traffic Engineering in Operational IP-Networks-An Experimental Study. In T. Magedanz, E. Madeira, and P. Dini, editors, *Operations and Management in IP-Based Networks*, pages 202–211, Barcelona, Spain, October 2005. Springer. LNCS 3751.

[2] H. Abrahamsson and M. Björkman. Robust traffic engineering using L-balanced weight-settings in OSPF/IS-IS. In *Broadnets*, Madrid, Spain, Sept 2009.

[3] D. Applegate and E. Cohen. Making intra-domain routing robust to changing and uncertain traffic demands: Understanding fundamental tradeoffs. In *Proc. ACM SIGCOMM*, pages 313–324, Karlsruhe, Germany, August 2003.

[4] S. Balon and G. Leduc. Combined intra- and inter-domain traffic engineering using hot-potato aware link weights optimization. In *Proc. ACM SIGMETRICS*, pages 441–442, Annapolis, MD, USA, Jun 2008.

[5] B. Fortz and M. Thorup. Internet Traffic Engineering by Optimizing OSPF Weights. In *Proc. IEEE INFOCOM*, pages 519–528, Israel, March 2000.

[6] B. Fortz and M. Thorup. Optimizing OSPF/IS-IS weights in a changing world. *IEEE Journal on Selected Areas in Communications*, 20(4):756–767, 2002.

[7] A. Gunnar and M. Johansson. Robust routing under BGP reroutes. In *Proc. IEEE GLOBECOM*, Washington DC, USA, November 2007.

[8] A. Gunnar and M. Johansson. Robust routing under statistical uncertainty: models and polynomial time algorithms. In *Proc. NGI 2009*, Aveiro, Portugal, July 2009.

[9] M. Johansson and A. Gunnar. Data-driven traffic engineering: techniques, experiences and challenges. In *Broadnets*, San Jose, Califonia, USA, October 2006.

[10] A. Nucci, S. Bhattacharyya, N. Taft, and C. Diot. IGP link weight assignment for operational tier-1 backbones. *IEEE/ACM Trans. Netw.*, 15(4):789–802, 2007.

[11] M. Pioro and D. Medhi. *Routing, Flow and Capacity Design in Communication and Computer Networks*. Morgan Kaufmann Publishers, 2004.

[12] K.G. Ramakrishnan and M.A. Rodrigues. Optimal routing in shortest-path data networks. *Bell Labs Technical Journal*, 6(1):117–138, 2001.

[13] M. Roughan, M. Thorup, and Y. Zhang. Traffic engineering with estimated traffic matrices. In *Proc. ACM Internet Measurement Conference*, pages 248–258, Miami Beach, Florida, USA, October 2003.

[14] R. Teixeira, N. Duffield, J. Rexford, and M. Roughan. Traffic matrix reloaded: The impact of routing changes. In *Proc. Passive Active Measurements*, Boston, Massachusetts, USA, April 2005.

[15] R. Teixeira, T. Griffin, G. Voelker, and A. Shaikh. Network sensitivity to hot potato disruptions. In *Proc. ACM SIGCOMM*, pages 231–244, Portland, Oregon , USA, August 2004.

[16] S. Uhlig, B. Quoitin, S. Balon, and J. Lepropre. Providing public intradomain traffic matrices to the research community. *ACM SIGCOMM Computer Communication Review*, 36(1):83–86, January 2006.

[17] H. Wang, H. Xie, L. Qiu, Y. Yang, Y. Zhang, and A. Greenberg. Cope: traffic engineering in dynamic networks. In *Pric. ACM SIGCOMM*, pages 99–110, Pisa, Italy, 2006.

[18] Y. Wang, Z. Wang, and L. Zhang. Internet traffic engineering without full mesh overlaying. In *Proc. IEEE INFOCOM*, pages 565–571, Anchorage, Alaska, USA, May 2001.

[19] D. Xu, M. Chiang, and J. Rexford. Deft: Distributed exponentially-weighted flow splitting. In *Proc. IEEE INFOCOM*, pages 71–79, Anchorage, Alaska, USA, May 2007.