

# IGP Link Weight Assignment for Transient Link Failures

A. Nucci<sup>a</sup> and B. Schroeder<sup>b</sup> and S. Bhattacharyya<sup>a</sup> and N. Taft<sup>a</sup> and C. Diot<sup>c\*</sup>

<sup>a</sup>Sprint Advanced Technology Laboratories  
1 Adrian Court, Burlingame CA 94010, USA

<sup>b</sup>Department of Computer Science  
Carnegie Mellon University, Pittsburgh PA, USA

<sup>c</sup>Intel  
15 JJ Thomson Avenue, Cambridge, UK

Intra-domain routing in IP backbone networks relies on link-state protocols such as IS-IS or OSPF. These protocols associate a weight (or cost) with each network link, and compute traffic routes based on these weights. However, proposed methods for selecting link weights largely ignore the issue of failures which arise as part of everyday network operations (maintenance, accidental, etc.). Changing link weights during a short-lived failure is impractical. However such failures are frequent enough to impact network performance.

We propose a Tabu-search heuristic for choosing link weights which allow a network to function almost optimally during short link failures. The heuristic takes into account possible link failure scenarios when choosing weights, thereby mitigating the effect of such failures. We find that the weights chosen by the heuristic can reduce link overload during transient link failures by as much as 40% at the cost of a small performance degradation in the absence of failures (10%).

## 1. Introduction

Large IP networks use a link-state protocol such as IS-IS or OSPF as their Internal Gateway Protocol (IGP) for intra-domain routing. Every link in the network is assigned a weight, and the cost of a path is measured as the sum of the weights of all links along the path. Traffic is routed between any two nodes along the minimum cost path which is computed using Dijkstra's shortest path forwarding (SPF) algorithm. Thus setting the link weights is the primary traffic engineering technique for networks running IS-IS or OSPF.

The traffic engineering objectives of an IP backbone are largely determined by the service-level agreements (SLAs) with its customers, which typically include speed-of-light delays and lossless packet delivery. Accordingly, there are two main goals when setting link weights: keeping end-to-end delays low and ensuring that no link is overloaded. Several formalizations of the problem of selecting link weights that are optimal with respect to the above traffic engineering objectives have been shown to be NP-hard [1,2]. A common recommendation of router vendors is to set the weight of a link to the inverse of its capacity [3]. The idea is that this will attract more traffic to high capacity links and less traffic to low capacity links, thereby yielding a good distribution of traffic load. Another common recommendation is to assign a link a weight proportional to its physical length in order to minimize propagation delays. In practice, many backbone operators

---

\*B. Schroeder and C. Diot were at Sprint ATL when this work was done.

use the ad-hoc approach of observing the flow of traffic through the network, and repeatedly adjusting the weight whenever the load on a link is higher or lower than desired. The problem has been addressed formally [1,4,5] using different techniques from operations research. Related work is discussed in more detail in the extended version of this paper [6].

A drawback of most current approaches (with the notable exception of [4]) is that they view the link weight assignment problem as a static problem largely ignoring network dynamics. However in practice, one of the main challenges of a network operator is to deal with link failures that are encountered on a daily basis in large IP backbones. When a link fails, IS-IS/OSPF routing diverts the traffic over that link to alternate paths, increasing the load of or more of the other links. The most obvious way of restoring the network to its original traffic engineering objectives is to perform a network-wide recomputation and reassignment of link weights. [4] has shown that changing just a few link weights is usually sufficient to rebalance the traffic. However changing link weights during a failure may not be practical for two reasons. First, the new weights will have to be flooded to every router in the network, and every router will have to recompute its minimum cost path to every other router. This can lead to considerable instability in the network, aggravating the situation already created by the link failure. The second reason is related to the short-lived nature of most of the link failures. In a study of inter-PoP link failures in Sprint's IP backbone over a four-month period, it was observed that as many as 80% of the failures last less than 10 minutes and 50% of the failures last even less than a minute [7,6]. We define transient failures as those failures that last less than 10 minutes. Transient failures can create rapid congestion that is harmful for the network. However, they leave a human operator with insufficient time to reassign link weights before the failed link is restored.

The study of inter-PoP link failures in the Sprint network shows that more than 70% of the transient failures (less than 10 minutes) are isolated[6]. For the long-lived failures (longer than 10 minutes), only about 30% are isolated. This is because a majority of the long-lived failures are caused by optical fiber cuts that can affect several links simultaneously. In contrast, transient failures are due to maintenance windows, line card and optical equipment failures, and router software problems.

We focus on minimizing the impact of transient link failures on traffic. Accordingly, we propose an approach for assigning link weights in IS-IS/OSPF networks that takes into account the isolated failure of any possible link in the backbone. The goal is to find one set of link weights that works well in the absence of failures and, at the same time, does not overload any link during transient failures. The heuristic achieves this goal by finding weights that balance traffic evenly among alternate paths during a link failure. This approach provides considerable benefit for stable network operations since the network can function almost optimally in the presence of transient failures without requiring any weight change. This is also the only applicable approach for the kind of failures addressed in this work.

We study the performance of this heuristic using a computation-based approach. For a given network topology and traffic demand matrix, we first use our heuristic to assign link weights. We then use Dijkstra's SPF algorithm to determine minimum cost routes, and compute performance metrics of interest for these routes. We validate the performance of the heuristic using a lower bound obtained from an Integer Linear Program (ILP) model.

Results show that the heuristic performs within 10% of the lower bound. It is not possible to find a single weight set that performs equally well both for transient failures and in the absence of failures. However, our algorithm finds a weight set that restricts link loads to less than 90% during any link failure. In comparison, a traditional weight selection approach that does not consider link failures results in a maximum link load of 135% for the same network. This improvement comes at the cost of a small increase in maximum link load in the absence

of failures. However, there is no degradation in end-to-end delays across the network due to congestion.

For ease of exposition we discuss this work with reference to IS-IS in this paper. Nevertheless, all the results are equally applicable to any other Internal Gateway Protocol including OSPF.

The rest of the paper is organized as follows. Section 2 formally states the problem addressed in this work. Section 3 describes the Tabu Search heuristic for the link weight selection problem. Section 4 analyzes the performance of the heuristic. Section 5 concludes the paper. The Integer Linear Program (ILP) model used to generate a lower bound for the heuristic has been omitted here due to space constraints and can be found in the extended version of this paper [6].

## 2. The Link Weight Selection Problem

The problem of computing IS-IS link weights can be formally stated as follows. We represent a network by an undirected graph  $G(\mathcal{V}, \mathcal{L})$  where  $\mathcal{V}$  corresponds to the set of nodes and  $\mathcal{L}$  corresponds to the links connecting the nodes. Each edge  $l \in \mathcal{L}$  has a bandwidth capacity  $c_l$  and integer ISIS weight  $w_l$ . Let  $L$  be the cardinality of the set  $\mathcal{L}$ . Let  $D$  be a traffic matrix representing the traffic demands such that  $d^{sd} \in D$  correspond to traffic demand between origin node  $s$  and destination node  $d$ . Henceforth we will refer to an origin-destination network node pair as an O-D pair. The traffic demand between an O-D pair  $s$  and  $d$  is routed along the minimum cost path from  $s$  to  $d$ , where the cost of a path is measured as the sum of the weights of the links along the path. If multiple paths exist with the same minimum cost, the traffic demand is split equally among all these paths. The problem is to choose a single set of link weights  $\mathcal{W} = (w_1, w_2, \dots, w_L)$  so as to optimize a given objective function. The objective function is typically selected to meet the traffic engineering goals of a network.

An important traffic engineering goal for IP networks is to distribute traffic evenly so that no link carries an excessive load. The load on a link is defined as the ratio of the traffic carried by the link to its capacity. A common approach is to choose routes so as to minimize maximum load over all links in the network. This is also the objective function chosen for this work. The problem of choosing link weights to minimize maximum link load is NP-Hard [1,2].

A novel aspect of our approach to the link weight selection problem is that we consider transient link failures. We model the occurrence of isolated transient failures in the network as state transitions. In the absence of any such failure, the state of the network is denoted by  $S_0$ . During the transient failure of link  $i$  ( $i = 1, 2, \dots, L$ ), the state of the network is denoted by  $S_i$ . We refer to the maximum link load of the the network in any given state as the *bottleneck load*. Let  $\mu(S_j) \geq 0$  be the bottleneck load in state  $S_j$ .  $S_{wst}$  is defined as the state with the maximum bottleneck load over all network states, i.e.,  $\mu(S_{wst}) \geq \mu(S_j), \forall j \in [0, L]$ . Henceforth we refer to state  $S_0$  as the no-failure state, and the state  $S_{wst}$  as the worst-failure state.

The problem of selecting IS-IS weights has been traditionally addressed for state  $S_0$ , which implies that link weights are selected to minimize  $\mu(S_0)$  [1,4,5]. Instead a network operator may be interested in limiting the maximum link overload due to *any* single-link failure event. In that case, it is appropriate to choose link weights so as to minimize  $\mu(S_{wst})$ . Selecting a weight set that is optimal for the network in a failure state may result in sub-optimal performance in the absence of failures. We want to meet two objectives: preventing link overloads during a failure while simultaneously minimizing performance degradation in the absence of failures. We thus choose the following objective function:

$$\mathcal{F} = (1 - W)\mu(S_0) + W\mu(S_{wst}) \tag{1}$$

where  $W \in (0, 1)$  is an artificial parameter that helps in minimizing performance degradation

in the absence of failures while maximizing the gain for any transient failure. Setting  $W$  to 0 corresponds to finding link weights without considering failures. On the other hand, setting  $W$  to 1 results in link weights that minimize the maximum load under any failure, but completely ignore performance in the absence of failures. For very small or very large values of  $W$  (e.g.,  $W = 0.0001$  or  $W = 0.9999$ ), the term in  $\mathcal{F}$  with the large coefficient is the major objective, yielding potentially several optimal solutions that are disambiguated by the value of the second term in  $F$ .

### 3. Search Heuristic for Link Weight Selection

The heuristic we propose is based on the Tabu Search (*TS*) [8] methodology. TS conducts a guided exploration of the space of admissible solutions, keeping track of all solutions evaluated along the way. The exploration starts from an initial solution that is generally obtained with a greedy algorithm. When a stop criterion is satisfied, the algorithm returns the best visited solution. To move from one solution to the next, TS explores the neighborhood of the last solution visited (referred to as the current solution). It generates a neighbor solution by applying a transformation, named as a *move*, on the current solution. The set of all admissible moves (determined by the rules guiding TS) uniquely defines the *neighborhood* of the current solution. At each iteration of the TS algorithm, all solutions in the neighborhood are evaluated, and the best is selected as the new current solution.

A special rule, the *Tabu list*, is introduced in order to prevent the algorithm from deterministically cycling among solutions already visited. The Tabu list stores the last accepted moves; while a move is stored in the Tabu list, it cannot be used to generate a new move. The choice of the Tabu List size is very important - too small a size may cause a periodic repetition of the same solutions, while too large a size may severely limit the number of applicable moves, thus preventing a good exploration of the solution space.

The fundamental elements of the link weight assignment heuristic, TabuISIS, are described below:

- Precomputation Step.** Before running the Tabu Search, we use an approximation to speed up the search procedure by reducing the size of the search space. Each solution in the search space consists of a set of routes for all traffic demands. We therefore choose a criterion to filter out a subset of possible routes. This filtering is based on a hop-count threshold. Only routes with hop-counts less than the threshold are considered admissible and considered during the search procedure. This approximation is based on the intuition that Tabu Search will generally avoid very long routes that consume network resources unnecessarily. The choice of a hop-count threshold is based on the network topology under consideration. Choosing a small value for the threshold will eliminate most routes from consideration and may prevent Tabu Search from visiting the optimal solution. On the other hand, a large value of the threshold may result in a very large search space. The time taken to find the optimal solution grows exponentially with the size of the search space, thereby making the search procedure very slow. Thus the hop-count threshold allows Tabu Search to efficiently explore the search space with a reasonable computational overhead.
- Initial Solution.** The choice of the initial solution is important since it can significantly affect the time taken by the search procedure to converge to the final solution. We set the initial weight of each link to be the inverse of the link capacity - a common recommendation of router vendors [3].

- Moves and Neighborhood Generation.** Each move corresponds to perturbing one or more of the weights in the current weight set. The first step in a move is to run Dijkstra's SPF algorithm for the current weight set, generate all minimum-cost routes and compute the traffic load for each link. We then identify two sets of links - links whose loads are within a small percentage of the maximum load (heavily loaded) and links whose loads are within a small percentage of the minimum load (lightly loaded). A link is selected at random from the heavily loaded set and its weight is increased (in order to divert traffic to other paths and reduce its load). Then, a link is selected at random from the lightly loaded link set and its weight is decreased. The goal is to attract traffic towards this link and potentially reduce the load on other, more heavily loaded, links. A new neighborhood is designed by repeating the above procedure.
- Evaluation of each solution.** Every solution in the neighborhood is evaluated as follows. Minimum cost routes are computed using the SPF algorithm and the traffic load on each link is determined for a given traffic matrix. This evaluation is performed when there is no failure in the network and for each possible link failure. Then the objective function in equation (1) is computed.
- Diversification.** This is needed to prevent the search procedure from indefinitely exploring a region of the solution space with only poor quality solutions. It is applied when there is no improvement in the solution after a certain number of iterations (each iteration consists of generating a new neighborhood and selecting the best solution in the neighborhood). The diversification is a modification of the move described earlier. In the case of a move, only one link each is chosen at random from the heavily loaded link set and the lightly loaded link set. For diversification, several links are picked from each set. The weights of the selected links from the heavily loaded set are increased while the weights of the selected links from the lightly loaded links are decreased. This diverts the search procedure to a different region of the solution space where it resumes the process of neighborhood generation and solution evaluation.
- Tabu List.** For TabuISIS, the Tabu list is used to remember the most recent moves made, consisting of the links whose weights have been changed and the increase/decrease applied to the link weight.
- Stop Criterion.** The search procedure stops when a fixed pre-determined number of iterations is reached. The number of iterations is defined based on the size of the network in consideration, the computational time needed and the quality of solution desired.

## 4. Results

We analyze the performance of our link weight assignment heuristic using a computation-based approach. For this evaluation, we choose a network topology and a traffic matrix and compute link weights using TabuISIS. Based on these weights, the minimum-cost routes for all source-destination traffic demands are determined and the resulting load on every link is calculated. The objective function value for this heuristic solution is then compared against the lower bound obtained from the ILP model (presented in [6]). The ILP is solved using the CPLEX software package [9].

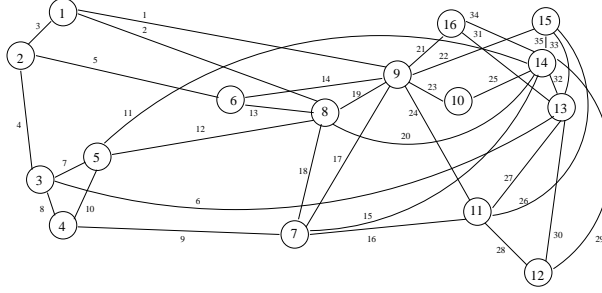


Figure 1. PoP-level Sprint network

#### 4.1. Topology

The primary goal is to understand the benefits and/or tradeoffs, if any, of taking single-link failures into account when computing IS-IS link weights. For IP backbones, the failure of long-haul inter-PoP links between cities is significantly more critical than link failures within a PoP, since every PoP has a highly meshed topology. Accordingly we consider a single representative topology based on the Sprint IP backbone, consisting of 16 nodes (each corresponding to a PoP) and 68 full-duplex links (Figure 1). Links are assigned integer weights in the range of 5 and 255, based on current practices in the Sprint network.

We emphasize here that TabuISIS does not rely on the properties of any specific network topology and is therefore equally applicable to any topology or graph.

#### 4.2. Traffic Matrices

The second input for the link weight selection problem is the traffic demands between origin-destination node pairs, represented by a traffic matrix. Unfortunately very little data exists on PoP-to-PoP traffic matrices for operational backbones. We therefore use synthetic traffic matrices for evaluating our heuristic. In this paper, results are presented for two different forms of traffic matrices:

- **Gravity Model (GM).** This form of the traffic matrix is based on the findings in [10] about the characteristics of PoP-to-PoP traffic matrices in Sprint's IP backbone. The volume of traffic from node  $i$  to node  $j$  is selected as follows:

$$d_{i,j} = O_i e^{V_j} / \sum_{j \in [1, V]} e^{V_j} \quad \forall i \in [1, V] \quad (2)$$

where  $O_i$  is the total volume of traffic originating at node  $i$ , given by

$$O_i = \begin{cases} \text{uniform}(10,50), & \text{if } prob \in [0, 0.6] \\ \text{uniform}(80,130), & \text{if } prob \in [0.6, 0.95] \\ \text{uniform}(200,300), & \text{if } prob \in [0.95, 1] \end{cases}$$

and  $prob$  is a uniform random variable between 0 and 1.

$V_j / \sum_{j \in [1, V]} e^{V_j}$  is the share of traffic originating at node  $i$  that is destined to node  $j$ . The  $V_j$  is a random number picked according to a uniform distribution between 1 and 1.5.

The key idea is to create three kinds of traffic demands by volume. The fan-out of traffic originating at a given PoP is then determined by equation (2) in accordance with the observations in [10].

- **Negative Exponential (NegExp).** Each entry in the matrix is generated according to a negative exponential distribution with mean 40 Mbps. This is one of the several other forms of traffic matrices that we evaluated in order to validate the generality of our results.

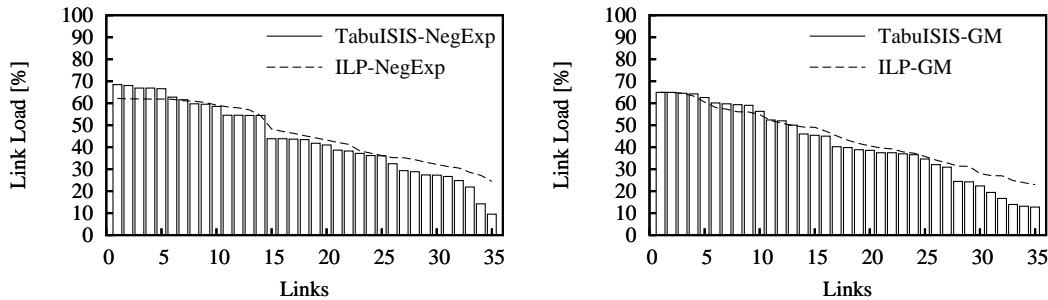


Figure 2. Comparison of link loads for TabuISIS and the optimal ILP solution with  $W = 0$  for the NegExp traffic matrix (left) and GM traffic matrix (right).

### 4.3. Reducing the size of the search space for TabuISIS

As mentioned in Section 3, the search space is reduced in the precomputation step using a hop-count threshold. Based on a careful analysis of TabuISIS [6], the hop-count threshold was chosen to be 8 for the Sprint network.

### 4.4. Validation of TabuISIS

We validate the quality of the TabuISIS solution by comparing it with the ILP model. For TabuISIS we find a set of link weights for the NegExp and GM traffic matrices, compute shortest path routes and the load on each link. For the ILP model, we generate the set of optimal routes and then compute the load on each link. Figure 2 shows the link loads for TabuISIS and the ILP model for both types of traffic matrices. TabuISIS link loads are shown as bars while the ILP solution link loads are shown as a dashed continuous line. Links are ranked on the x-axis by decreasing link loads under TabuISIS. These results were obtained with  $W = 0$ , which corresponds to the traditional objective of minimizing maximum load in the absence of failures.

For the NegExp traffic matrix, the maximum link load is 68% for TabuISIS and 62% for the ILP solution - a difference of only 6%. Moreover, the distribution of link loads in the two cases are similar. For the GM traffic matrix, both TabuISIS and the ILP model yields the same value for the maximum link load. This indicates that TabuISIS was able to find the optimal value of maximum link load achievable by any routing strategy (with respect to the given objective function). This case has a special significance which is discussed later. The distribution of link loads for TabuISIS and the ILP solution are also similar for the GM traffic matrix.

Results for the no-failure case ( $W = 0$ ) are shown for a specific reason. We will subsequently evaluate the benefits of considering link failures in weight selection by comparing against weight selection with  $W = 0$ . Therefore, it is important to establish first that TabuISIS yields a solution of high quality for  $W = 0$ . However, we have performed this comparison for all values of  $W$  in  $[0,1]$  and found that the difference in the value of the objective function in equation 1 is less than 10% in all cases.

### 4.5. Considering link failures in weight selection

Figure 3 shows the variation in maximum link load with  $W$  for the NegExp traffic matrix. The maximum link loads are shown for the no-failure state (solid line) and the worst-failure state (dotted line). A value of  $W = 0$  corresponds to a set of weights picked without considering failures.  $W = 0$  corresponds to selecting weights that optimize performance in the no-failure state but ignore the performance degradation during failures. On the other hand,  $W = 1$  corresponds to optimizing performance in the worst-failure state while ignoring performance in the no-failure state. We find that when  $W$  is varied from 0 to 1, the maximum link load in the no-failure state increases from 68% to 80%. On the other hand, the maximum link load in the

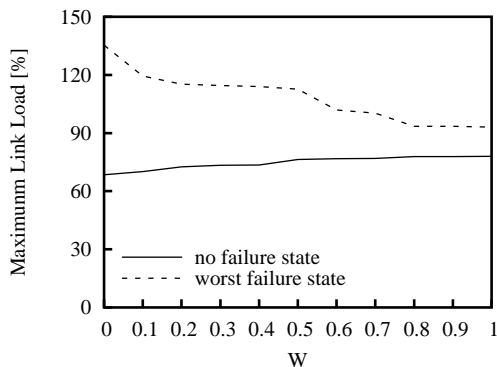


Figure 3. Variation in maximum link load with  $W$  for two cases - no-failure and the worst-failure.

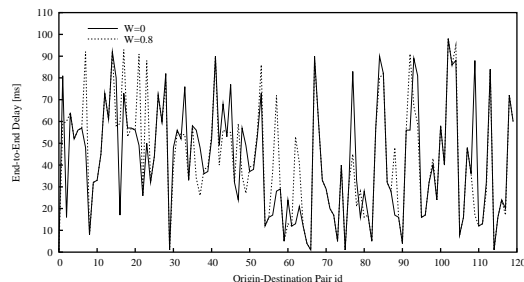


Figure 4. End-to-end delays per origin-destination pair for link weights chosen with  $W = 0$  and  $W = 0.8$ .

no-failure case reduces from 135% to 90%. This indicates that that performance during failures can be significantly improved by considering failures in link weight selection. However, this is achieved at the cost of a small performance degradation in the no-failure state.

The maximum link load in the worst-failure state stays the same between  $W = 0.8$  and  $W = 1$ . However, the maximum link load for the no-failure state increases slightly between  $W = 0.8$  and  $W = 1$ . Therefore  $W = 0.8$  produces the optimal performance trade-off between the no-failure state and the worst-failure state for the Sprint network and the NegExp traffic matrix. However, the most suitable value of  $W$  depends on the combination of topology and traffic matrix. It is therefore important to do the above analysis instead of picking a random value in  $[0, 1]$  or just using the boundary values.

Service-level agreements between network operators and customers include bounds on the average end-to-end delay. Therefore it is important to ensure that our approach of considering failures in weight selection does not degrade end-to-end delays in the network. Figure 4 shows the end-to-end delay for each origin-destination node pair for two sets of link weights - one chosen without considering link failures (solid line) and the other chosen by considering link failures (dotted line). These delay values were computed using knowledge of link propagation delays in the Sprint network. We observe that there are significant differences for a few O-D pairs, e.g., for O-D pair 7, the delay increases from 48 milliseconds to 92 milliseconds when link failures are considered in weight selection. However, the overall difference is small: the mean and standard deviation are 42.22 msecs and 26.23 msecs when failures are not considered in weight selection, and 42.93 msecs and 26.02 msecs when they are. Moreover, the maximum delay over all O-D pairs is the same in both cases (93 msec).

We now examine more closely some of the changes in weight assignment and route selection for  $W = 0$  and  $W = 0.8$ . Figure 5 shows the distribution of link loads for weight sets selected with  $W = 0$  and  $W = 0.8$  for the NegExp traffic matrix. In each graph, the solid bars show the link load in the no-failure state. The dashed bar for each link shows the maximum load of this link for any single link failure. Link Ids on the x-axis correspond to the labels on links in Figure 1. The distribution of link loads in the no-failure state is very similar in both cases. The maximum load for any single failure reduces from 135% for  $W = 0$  to 90% for  $W = 0.8$  (as seen earlier in Figure 3). Moreover, the distribution of maximum link load for any single failure has less variation for  $W = 0.8$ .

To gain insights into why it helps to consider link failures in weight selection, we focus on some specific links. Figure 6 shows the link weights for  $W = 0$  in solid bars, and  $W = 0.8$  in dashed bars. Link 23 connecting nodes 9 and 10 (Figure 1) and link 25 connecting nodes 10

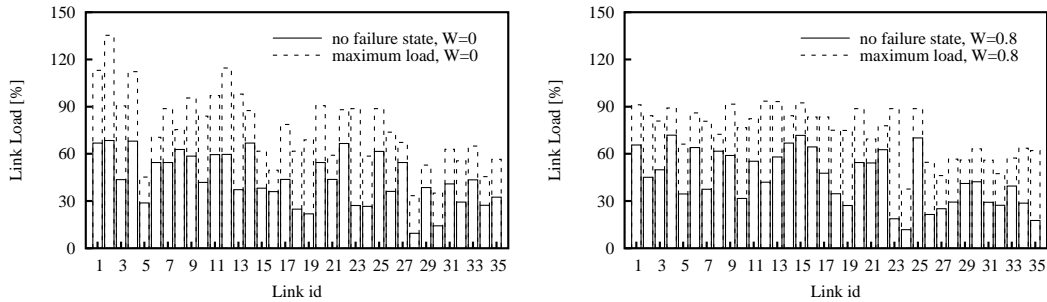


Figure 5. Link loads for weights selected with  $W = 0$  and  $W = 0.8$  for the Sprint network and NegExp traffic matrix. The solid bars show the loads in the absence of failures. The dashed bar for each link shows the maximum load on that link for any link failure.

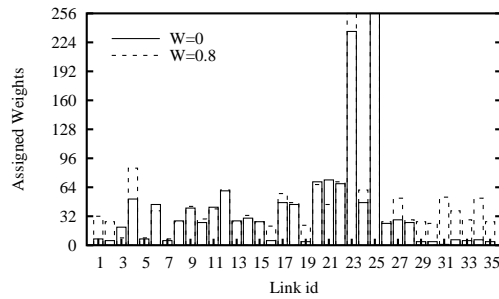


Figure 6. Link weights selected with  $W = 0$  and  $W = 0.8$  for the NegExp traffic matrix.

and 14 are assigned very large weights for both  $W = 0$  and  $W = 0.8$ . These links are the only ones connecting node 10 to the rest of the network. When one fails, the other has to absorb all the load exchanged between node 10 and the rest of the network. For  $W = 0$ , TabuISIS does not consider this failover scenario when selecting weights. But it assigns large weights to these links in order to ensure that transit traffic (i.e., traffic not destined to node 10) is diverted away from these links. With  $W = 0.8$ , TabuISIS actually considers the failover scenario and tries to increase the weights even further to mitigate overload in the event of failure of either link 23 or 25.

The link that carries the maximum load for any single-link failure is link 2 connecting nodes 1 and 8. This maximum load is caused by the failure of link 1 which connects nodes 1 and 9. Node 1 is connected to the rest of the network via three links - link 1 to node 9, link 2 to node 8 and link 3 to node 2. For  $W = 0$ , TabuISIS assigns weights 7, 5 and 20 respectively to links 1, 2 and 3. In the absence of failures, the loads on these links are well-balanced - 67%, 68% and 43% respectively. However, when link link 1 fails, traffic between node 1 and 13 of the 15 other nodes fail over to link 2 resulting in a load of 135%.

When TabuISIS considers link failures ( $W = 0.8$ ), it actually plans ahead for the possible failure of link 2 and assigns weights 32, 26 and 8 to links 1, 2 and 3 respectively. When link 1 fails, traffic between node 1 and 8 of the 15 other network nodes fails over to link 2, while the traffic between node 1 and the rest of the nodes fails over to link 3. This balances the failover traffic on the two alternate paths and reduces the impact of failure on any single link.

The extended version of the paper contains results on two additional topics - how topology limitations affect weight selection strategy, and how the range of link of weights affects performance. For details, the interested reader is referred to [6].

## 5. Conclusions and New Directions

We have examined the problem of link weight assignment for short isolated link failures, and proposed a novel Tabu Search Heuristic that takes into account such failures. To the best of our knowledge this is the first work to consider transient link failures for link weight selection. The approach is likely to be attractive to IP backbone operators who encounter short link failures on a daily basis.

The proposed heuristic finds a single weight set that performs well in the absence of failures but prevents severe link overloads during failures. Results indicate that the heuristic finds a solution that is within 10% of the optimal lower bound. The maximum load on a link during any single-link failure can be reduced by as much as 50% over an approach that does not consider failures in weight selection. This improvement comes at the cost of a small performance degradation (10%) in the absence of failures.

This paper opens up a new avenue for studying the link weight assignment problem and the interaction of routing with link failures. Both the heuristic and the ILP model [6] are based on the notion of “network states” where each state corresponds to either no failure or a single link failures. Therefore this notion can be easily extended to the case of multiple link failures. For example, a network may have critical groups of links which are likely to fail together because they share common optical fiber conduits. In that case, each network state corresponds to the failure of a critical group. The heuristic as well as the ILP model can be applied to such multiple link failure scenarios with relatively straightforward extensions. In addition, work remains to be done on how errors in the input traffic matrix may affect link weight selection, and how often link weights have to be updated to track changes in the traffic matrix.

## REFERENCES

1. B. Fortz and M. Thorup. “Internet Traffic Engineering by Optimizing OSPF Weights”, *Proceedings of IEEE Infocom*, Tel-Aviv, Israel, March 2000.
2. M. Pioro, A. Szentesi, J. Harmatos, A. Juttner, P. Gajowniczek and S. Kozdrowski. “On OSPF Related Network Optimisation Problems”, *Proceedings of IFIP ATM IP*, Ilkley UK, July 2000.
3. D. Oran (Editor). “OSI IS-IS Intradomain Routing Protocol”, IETF Request For Comments 1142.
4. B. Fortz and M. Thorup. “Optimizing OSPF/IS-IS Weights in a Changing World”, *IEEE JSAC Special Issue on Advances in Fundamentals of Network Engineering*, 2001.
5. J. Harmatos. “A heuristic algorithm for solving the static weight assignment optimisation problem in OSPF networks”, *Proceedings of Global Internet Conference*, Dallas USA, Nov 2001.
6. A. Nucci and B. Schroeder and S. Bhattacharyya and N. Taft and C. Diot. “IGP Link Weight Assignment for Transient Link Failures”, *Sprint ATL Technical Report TR02-ATL-071000*.
7. G. Iannaccone, C. Chuah, R. Mortier, S. Bhattacharyya and C. Diot. “Analysis of Link Failures in a Large IP Backbone”, *Proceedings of 2nd ACM Sigcomm Internet Measurement Workshop (IMW)*, San Francisco USA, November 2002.
8. F. Glover and M. Laguna. “Tabu Search”, *Kluwer Academic Publishers*.
9. <http://www.ilog.com/products/cplex/>.
10. A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya and C. Diot. “Traffic Matrix Estimation: Comparisons and New Directions”, *Proceedings of ACM SIGCOMM Pittsburgh USA*, August 2002.