The Interaction of IGP Weight Optimization with BGP

Selin Cerav-Erbas
Dept. of Management Science
Université catholique de Louvain, Belgium

Olivier Delcourt
Research Unit in Networking
Université de Liège, Belgium

Bernard Fortz
Dept. of Management Science
Université catholique de Louvain, Belgium

Bruno Quoitin
Dept. of Computer Science and Engineering
Université catholique de Louvain, Belgium

Abstract

Link weight optimization is shown to be a key issue in engineering of IGPs using shortest path first routing. The IGP weight optimization problem seeks a weight array resulting an optimal load distribution in the network based on the topology information and a traffic demand matrix. Several solution methods for various kinds of this problem have been proposed in the literature. However, the interaction of IGP with BGP is generally neglected in these studies. In reality, the optimized weights may not perform as well as expected, since updated link weights can cause shifts in the traffic demand matrix by hot-potato routing in the decision process of BGP. Hot-potato routing occurs when BGP decides the egress router for a destination prefix according to the IGP lengths. This paper mainly investigates the possible degradation of an IGP weight optimization tool due to hot-potato routing under a worst-case example and some experiments which are carried out by using an open source traffic engineering toolbox. Furthermore, it proposes an approach based on robust optimization to overcome the negative effect of hot-potato routing and analyzes its performance.

1 Introduction

Autonomous systems (ASes) have a heterogeneous structure, i.e., they operate with various protocols each of which serves another purpose. Internet Service Providers (ISPs) consult on traffic engineering (TE) tools in order to measure, simulate their network and improve its performance under resource constraints. However, most of the TE tools focus on only one protocol and may neglect its interaction with others. The main purpose of the paper is to show how neglecting this interaction can decline the performance of a traffic engineering tool by focusing on a particular case, namely the interaction of the Interior Gateway Protocol (IGP) weight optimizer with Border Gateway Protocol (BGP).

Routing decisions in a domain are grounded on two main protocols, IGP and BGP. IGP inside an AS is responsible for routing the traffic between any two nodes in the system. Most popular IGPs such as OSPF [16] and IS-IS [3] are based on shortest path first routing where traffic flows are sent along shortest paths according to administrative weights assigned to links. Determination of the optimal IGP weights for a balanced network has been a hot topic in the current literature and many approaches to the variants of this problem have been proposed. The studies in [2], [6], [7], [9], [13] build a sample.

In contrast to IGP, BGP [20] in a domain facilitates the communication of border routers with other ASes in order to gather reachability information to outside destinations and learn AS-level paths. The border routers share this information with other routers in the AS by BGP. A router inside the domain forwards the interdomain traffic according to the evaluation of BGP learned routes with regard to some hierarchical rules. One level of the rule-steps is based on the comparison of the IGP lengths to the egress routers having the same quality. When the interdomain traffic is forwarded according to the IGP distances, the case is called as *hot-potato routing*.

Hot-potato routing and its effect on the network's performance have been previously investigated in various studies. In [22], a mechanism to evaluate the hot-potato changes at the router level is introduced. The mechanism is then implemented in an operational network. It has been shown that hot-potato routing can cause significant changes in BGP updates. In [23], an analytical model is introduced to capture network sensitivity to hot-potato routing. As stated in this study, changes in IGP configuration, due to link/node failures and updates in the weights, may affect BGP routes by hot-potato routing and cause shifts in BGP traffic. Continuously, shifts in BGP traffic alters the *traffic (demand) ma-*

0-7695-2649-7/06/$20.00 (c) IEEE

trix of the network representing the amount of traffic flow between any two routers. In [25], Uhlig implemented the sensitivity model of [23] for an operational network. In [18], a publicly available BGP simulator[1] is used to measure the effect of hot-potato routing under link/router failures which then aids the network administration in locating the routers/links whose failure change the selected BGP routes.

The IGP weight optimization (IGP-WO) problem seeks an optimal weight setting to balance the load over the network for a given traffic matrix. In reality, updating the weight setting with the optimal one may further alter the traffic demand matrix due to hot-potato routing. When the changes in the traffic matrix are large enough to make the optimal weights undesirable for the new state of the network, the optimizer is needed to re-run.

Hot-potato routing has been neglected in many of the previous IGP-WO studies. The authors know only one paper which focuses on the interaction of IGP-WO with BGP. In [1], Agarwal et al. performed some experiments on the Sprint IP [21] network to measure the impact of hot-potato routing on the performance of IGP and the impact of IGP-WO on the performance of BGP.

In order to overcome the negative effect of hot-potato routing they proposed to implement the weight optimizer on a point-to-multipoint traffic matrix. They showed that this approach smoothened the negative impact of hot-potato routing at the expense of longer running times. This paper also exploits the interaction of BGP and IGP-WO in the operational GÉANT network [11]. The main contribution of this paper is to show that when TE is seen as a process, the lack of consideration of hot-potato routing may cause a convergence failure. This paper also proposes an approach based on robust optimization in order to overcome the convergence problem and test it in the GÉANT network.

2 Hot-potato Routing

BGP provides reachability information among domains besides coordination and implementation of the routing policies of the domain. Routing within BGP is based on a *decision process*, evaluation of all known routes according to some hierarchical rules. The routers on the border of the domain exchange reachability information with the neighbor ASes via *external* BGP (eBGP) messages. Then they evaluate all of the routes learned by eBGP messages and share the best ones for each destination prefix with other routers in the domain by *internal* BGP messages. Thus, a router inside a domain may be informed of the available routes to a destination prefix both by eBGP and iBGP messages.

[1]The studies here also utilize this simulator which is explained in details later in the paper.

A router implements the following decision steps to select the egress router to forward BGP traffic [5], [10]:

1. Prefer routes with the highest local preference which reflects the routing policies of the domain.
2. Prefer routes with the shortest AS-level path.
3. Prefer routes with the lowest origin number, e.g., the routes originating from IGP are most reliable.
4. Prefer routes with the lowest MED (multiple-exit discriminator) type which is an attribute used to compare routes with the same next AS-hop.
5. Prefer eBGP learned routes over iBGP learned ones.
6. Prefer the route with the lowest IGP distance to the egress point.
7. If supported, apply load sharing between paths. Otherwise, apply a domain-dependent tie-breaking rule, e.g., select the one with the lowest egress ID.

Hot-potato routing occurs when the egress router to forward the BGP traffic is selected according to the IGP distances. Thus, changes in IGP configuration have an impact on how BGP traffic flows through the network. IGP configuration can be altered due to link/node failures and updates in link weights.

3 IGP Weight Optimizer

Given the topology information and the traffic matrix, the IGP-WO problem aims at determining the link weights for an optimal network performance. In this study, the optimization algorithm proposed by Fortz and Thorup in [9] is utilized. Their experiments with real and synthetic networks show that the tool can obtain a network congestion within a few percent of the lower bound. And with optimized[2] weights, the network can accept $50\% - 110\%$ more traffic than with Cisco's default weights proportional to inverses of capacities.

One of the key features of the optimizer is its objective function. A special function is assigned to each link in the network where it issues a penalty increasing piece-wise linearly with the total load on the link. The main target in using such a function is to avoid congestion in the network. Since it overcomes bottlenecks, this function is preferred to minimizing the maximum utilization[3] in the network, which is the most common way to measure congestion. A bottleneck can occur when the maximum utilization rate is dominated

[2]Although the algorithm is a heuristic, we will refer the output weight settings as "optimal" during the paper.

[3]The utilization of a link is determined by the ratio of the load to its capacity.

by a traffic of high volume between any two nodes with few possible paths.

Thus, the tool optimizes a cost function in the following form.

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

where $l_a(TM)$ and c_a represent the total load of the link a under the given traffic matrix TM and its capacity, respectively. The piece-wise increasing shape to the cost function is given with the following derivatives. As the load on the link increases, it becomes more costly to assign any additional load to it.

$$\phi^{'}(l_a(TM), c_a) = \begin{cases} 1 & \text{for } 0 \leq l_a/c_a < 1/3 \\ 3 & \text{for } 1/3 \leq l_a/c_a < 2/3 \\ 10 & \text{for } 2/3 \leq l_a/c_a < 9/10 \\ 70 & \text{for } 9/10 \leq l_a/c_a < 1 \\ 500 & \text{for } 1 \leq l_a/c_a < 11/10 \\ 5000 & \text{for } 11/10 \leq l_a/c_a < \text{inf} \end{cases}$$

Please, note that $\phi(0, c_a) = 0$.

In this tool, it is assumed that the traffic flow between any two nodes is splitted approximately equally among the outgoing links of the source which belong to at least one of the shortest paths to the destination. This practice is called as equal-cost multipath [16] and is applied widely in real networks.

For the search algorithm, the optimizer utilizes tabu search [12] which is a well known meta-heuristic technique. The details are skipped here. Interested readers are pointed to [9] and [14].

4 Interaction of IGP-WO with BGP

The BGP/IGP-WO interaction is two-sided; both of them may affect each other's performance negatively. When the weights in the network are updated with the optimal ones, the BGP routing tables, keeping all the paths learned to each known destination prefix, need to be updated. This process brings an overhead to BGP. On the other hand, if the changes in the routing tables trigger large shifts in the traffic matrix, the new weight setting may remain so undesirable that the optimizer is needed to run again. With a small example we will show that *a standard TE process*, whose steps are given below, can fail to converge in the worst case.

Step1: Given the initial traffic matrix TM_0, run the IGP weight optimizer to obtain the output weights w_0. $k \leftarrow 0$.

Step2: Install w_k to the network. If the new traffic matrix TM_{k+1} is the same as TM_k, then stop. Otherwise go to Step3.

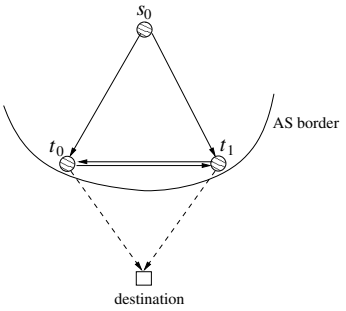

Figure 1. An example network.

Step3: $k \leftarrow k + 1$. Run the IGP-WO tool to obtain w_k. If w_k is the same as w_{k-1}, stop. Otherwise go to Step2.

In Figure 1, an example network with 1 ingress node (s_0), and 2 egress nodes (t_0, t_1) is given. There are 4 uni-directional links with transmission capacities of $1\ unit/sec$. We assume that there is a demand of $1\ unit/sec$ from s_0 to a given destination prefix and all of egress nodes have equal quality AS-level paths, i.e., BGP decides the next egress router according to IGP lengths. In this example, it is assumed that the IGP applies equal-cost multipath, in other words, the traffic flow between any source and destination is split approximately equally among the outgoing links which are on any shortest path in-between.

For the simple example, we select an initial traffic demand matrix which is a possible projection of the flow towards the given destination prefix. Let's assume, the initial traffic matrix starts with the flow sent over t_0:

$$TM_0 = \begin{array}{c} \\ (\end{array} \begin{array}{cc} t_0 & t_1 \\ 1 & 0 \end{array} \begin{array}{c} \\ ) \end{array}$$

When the effect of hot-potato routing is not taken into consideration during the optimization process, the IGP-WO and BGP interaction will follow these iterations:

Iteration 1: Given TM_0, the IGP-WO tool optimizes the network by assigning equal costs to both paths towards t_0. Thus, the flow is shared between the paths $(s_0 \rightarrow t_0)$ and $(s_0 \rightarrow t_1 \rightarrow t_0)$. With the following weight setting, the optimizer assumes a max utilization of 50% in the network.

(s_0, t_0)	(s_0, t_1)	(t_1, t_0)	(t_0, t_1)
2	1	1	1

With the new weights, the egress with the shortest distance is t_1. Thus, BGP decision process will end up with the following traffic matrix in the next iteration causing a max utilization of 100%.

$$TM_1 = \begin{array}{c} \\ (\end{array} \begin{array}{cc} t_0 & t_1 \\ 0 & 1 \end{array} \begin{array}{c} \\ ) \end{array}$$

Iteration 2: As in *Iteration 1*, the IGP-WO tool splits the demand equally between the paths $(s_0 \rightarrow t_1)$ and $(s_0 \rightarrow t_0 \rightarrow t_1)$. Thus, the output of the IGP-WO tool is given in the next table.

(s_0, t_0)	(s_0, t_1)	(t_1, t_0)	(t_0, t_1)
1	2	1	1

With the given weight settings, BGP responds to the updated weight settings by directing the traffic from s_0 to t_0. Thus, the traffic matrix will be in the following form:

$$TM_2 = TM_0 = \begin{array}{c} \begin{array}{cc} t_0 & t_1 \end{array} \\ \begin{pmatrix} 1 & 0 \end{pmatrix} \end{array}$$

Iteration 3: The TE system returns to the same conditions as in *Iteration 1* and fails to converge.

The negative effect of BGP over IGP optimization tool is not restricted with this convergence problem. With BGP, the network may obtain a performance much worse than the one that IGP traffic engineering tools assume they provided. As in this small example, users of the IGP-WO tool assume that the network will achieve a nice load balancing with a maximum utilization of 50%. However the reality may be far away from that.

5 Robust Optimization of IGP Weights

To overcome the convergence problem in the TE process, a robust optimization technique is proposed. Robust optimizers provide solutions which are *not necessarily optimal* but perform *well* for different cases of the system. The robust version of the IGP weight optimizer for multiple traffic matrices has been previously proposed in [8]. The robust IGP weight optimizer takes k $(k > 1)$ traffic matrices as input and minimizes

$$\min \Phi(TM_1, \ldots, TM_k) = \sum_{i=1}^{k} \phi(TM_i).$$

The *robust TE process* consists of the following steps:

Step1: Given the initial traffic matrix TM_0, run the IGP weight optimizer to obtain the output weights w_0. $k \leftarrow 0$.

Step2: Install w_k to the network. If the new traffic matrix TM_{k+1} is equal to one of $\{TM_0, \ldots, TM_k\}$, then stop. Otherwise go to Step3.

Step3: $k \leftarrow k + 1$. Run the robust IGP-WO tool for $\{TM_0 \ldots TM_k\}$ to obtain w_k. If w_k is the same as w_{k-1}, stop. Otherwise go to Step2.

Table 1. An example for robust optimization

	(s_0, t_0)	(s_0, t_1)	(t_1, t_0)	(t_0, t_1)	$\Phi(TM_0, TM_1)$
w_0	2	1	1	1	13.17
w_1	1	1	1	1	21.34
w_2	1	2	1	1	13.17

At each iteration the robust TE process optimizes simultaneously all of the traffic matrices created so far. The process stops when the newly appearing traffic matrix after a weight update is equal to one of the previous ones. At that point, the optimizer is not needed to run again, since this traffic matrix has been already optimized.

The robust TE process theoretically needs a finite number of steps to come to an end. In the worst case all of the possible traffic matrices can be issued during the whole run. However, in practice for the sake of stability, weight changes are avoided unless they provide considerable gains to the network. Thus, the TE process stops earlier, when the output weight setting causes a slight increase in the network performance.

We apply the robust TE process to our example. The process starts with the same initial matrix, TM_0.

Iteration 1: This step is similar to the one from the standard traffic engineering system. After the BGP decision process, the new traffic flow in the network becomes as in TM_1.

Iteration 2: Three potentially optimal link weight settings (w_0, w_1 and w_2) and the corresponding objective function values are given in Table 1. There are two alternatively optimal solutions: w_0 which is currently installed in the system and w_2. If we choose w_0, there is no need to update the weights. If w_2 is chosen, the new matrix will be the same as TM_0. According to Step2, in this case the process stops, too. In both cases, the TE process converges. Since weight changes are avoided, the network administrator would prefer the former.

6 Experimental Framework

The primary intent of the experimental work is to measure the effect of the interaction of IGP-WO with BGP on the TE process in real-networks. For this purpose, we utilize an open source traffic engineering software, called TOTEM [24], [15] and carry out some experiments on the GÉANT European research network.

The GÉANT network [11] is a multi-gigabit network, representing 30 countries and connecting 26 research and educational networks. The topology is composed of 23 nodes and 76 uni-directional high capacity links. In order to strengthen the need for traffic engineering, in the exper-

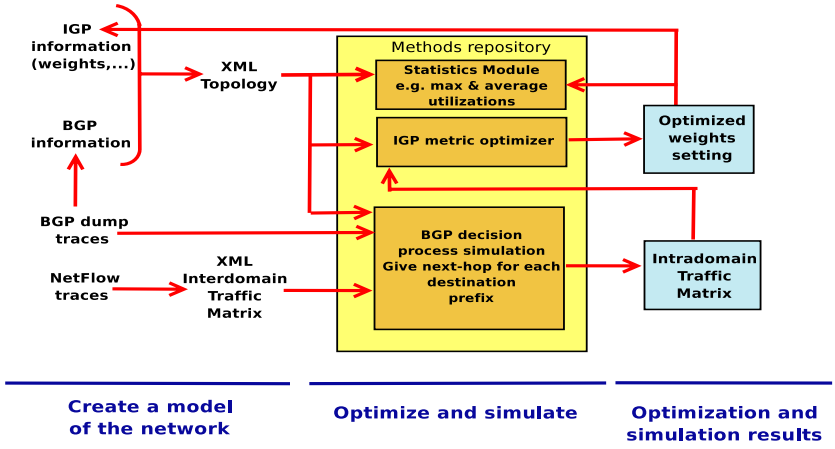

Figure 2. The Traffic Engineering Processes.

iments we divided the capacities of its 68 links by two. In its new form, the optimization within the network became more challenging.

We have obtained the NetFlow data and BGP dump traces for a day in August 2005 collected on each router of GÉANT. Basically, NetFlow data contains information about flows passing throw the network. NetFlow data is dumped every 15 minutes. Flows are cut by NetFlow timer (120 sec). Every flow is recorded by the ingress node, i.e. the node by which the flow enters GÉANT. The BGP traces contain daily dumps of all the routes towards the destination prefixes. These routes are gathered by a monitoring machine which collects all the exchanged BGP messages.

6.1 The TOTEM Toolbox and C-BGP

TOTEM [24], [15] consists of various tools to measure, control, simulate and optimize the network. It can be deployed both in on-line and off-line modes. The on-line deployment complements existing Network Management Systems (NMS) and provisioning tools. The possible off-line deployment of TOTEM is a simulator mode where the toolbox is simply connected to some monitoring tools to collect the data. Simulated data can also be produced internally. In the off-line mode, the user is typically expected to write some "what-if" scenarios based on the existing TE methods, execute them and collect the results. In this study, the TOTEM toolbox is deployed in the simulator mode.

C-BGP[4], [19] is an open source stand-alone BGP decision process simulator which is also integrated into TOTEM. Since it does not simulate the BGP message transmissions in packet-level, it is quite fast and efficient. It has two main inputs: the topology information with IGP events (also supports IS-IS) to calculate shortest path trees and BGP MRT dumps [17] to gather the interdomain paths learned by BGP from the neighbor domains. C-BGP can be configured with a CISCO-like syntax. According to the configuration it returns the selected interdomain paths for each router towards each destination prefix.

Since there are about 150000 prefixes, which is huge to replay in C-BGP, a clustering technique is used to shorten the running time. The prefixes announced in the same way (i.e., on the same GÉANT node, from the same peer, with the same BGP parameters) are put in the same cluster. Only one prefix from each cluster is announced in C-BGP. This allows us to advertise about 400 prefixes, called *clustered destination prefixes*, into the simulator. In order to project the egress node of a destination prefix on an ingress node, we need to find the corresponding advertised prefix of its cluster. Then, we retrieve the routing table of the ingress node and determine the egress node for the prefix.

6.2 Traffic Engineering Processes

The implementation of the TE processes from data collection to performance analysis report is described in Figure 2. The first step is to collect the IGP and BGP information and aggregate them to produce a topology file. Based on the NetFlow traces, the toolbox creates an interdomain traffic matrix, representing the amount of flow from the ingress routers to the clustered destination prefixes. The BGP dump traces, the interdomain traffic matrix and the topology file build the input to C-BGP. The BGP decision process simulator outputs the intradomain traffic matrix by projecting the destination prefixes on the egress routers. The IGP weight optimizer returns a weight array according to the topology information and intradomain traffic matrix. After the network is updated with the output weight setting, the statistics module returns information about the network's performance, e.g., the maximum and average uti-

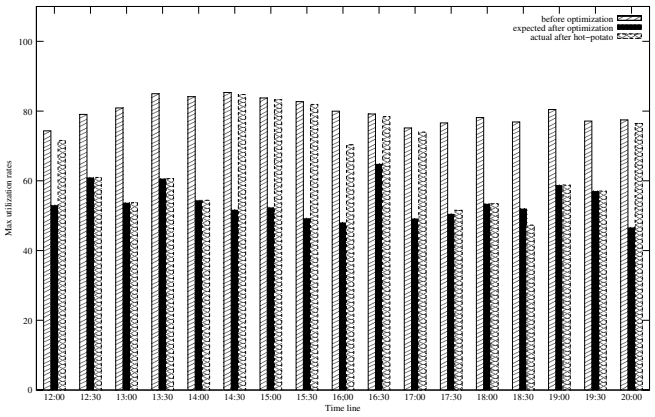

Figure 3. Hot-potato effect on max. utilization rates

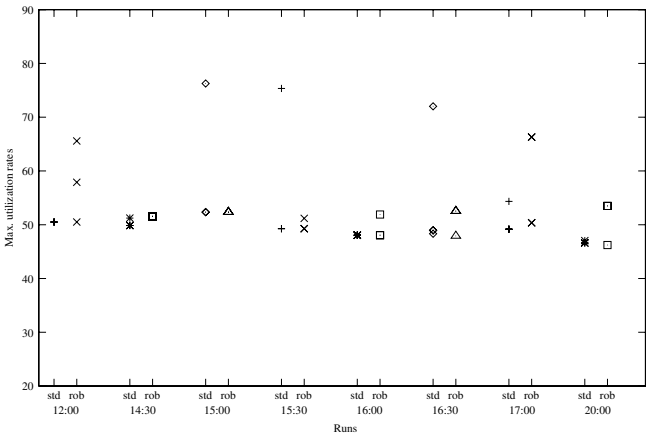

Figure 4. Hot-potato effect on max. utilization rates

lization rates, the value of the piece-wise linear objective function value. These steps repeat themselves, until the stopping conditions are met. During each specific run, we assume that the traffic flows do not change with time and the traffic matrix changes only due to hot-potato routing.

Both the standard and robust TE processes follow the steps in Figure 2. They differ in terms of the inputs to the IGP-WO tool and the stopping criteria. The stopping conditions given in the definitions of TE processes are theoretical, especially the former stopping condition remain too strict in practice. Since updating the weight changes brings an overhead to the networks, this stopping condition is modified to become more realistic. Both TE processes stop when the optimizer returns weights which do not cause a reasonable advantage in terms of the objective function (e.g., a decrease less than 2%) for the running traffic matrix.

7 Results

The standard TE process is implemented for the traffic matrices obtained from the NetFlow dumps gathered between 12:00 and 20:00 at 30 minutes intervals. For all of the runs, the maximum possible weight value and the number of iterations in the IGP-WO tool are selected as 50 and 5000, respectively. The optimizer has a preference option for the initial weight array of the tabu search algorithm: randomly generated or currently used. In our runs, the first weight optimization task inside a single TE process starts with a weight array created randomly, whereas the following ones start with the weights installed currently in the network.

Initially, we investigate the frequency of the negative effect of hot-potato routing on the network performance. In Figure 3 we plot the maximum utilization rates before the first optimization process with the expected and actual ones after the implementation of optimal weights. For example, in Figure 3 it can be seen that before the first optimization the network has a maximum utilization rate of 74.31% with the traffic matrix obtained from the NetFlow traces dumped at 12:00. According to the optimization process, it is expected that the maximum utilization rate would decrease to 53.01%. However, due to hot-potato routing, the max utilization becomes 71.55% with the new traffic matrix. Thus, the network performs much worse than expected. This type of situation occurs 8 times out of 17 runs. The worst effect of hot-potato routing in terms of maximum utilization rate occurs at 15:30 where the actual rate is 66% more than the expected.

During these experiments, we did not observe any example where the standard TE process failed to converge. Except the eight cases, the standard TE processes stop without the need for implementing the optimal weights proposed by the second optimization, i.e., the optimal weights propose an improvement less than 2% in terms of objective function. The details regarding both the robust and standard processes for the 8 cases are given in Table 2 for performance comparison. Each row in the table corresponds to a specific run, for example the first row gives the objective function values and maximum utilization rates observed in the network and expected with the optimal weights during the standard TE process carried out for the NetFlow data dumped at 12:00. In this table, we skipped the details regarding the initial condition of the network and the first optimization process, since they are already given in Figure 3. The cells written in bold give the resulting condition of the network for each run. Our first observation is that both types of TE needs the same number of optimization steps to stop, except the runs at 14:30, 16:00, 16:30. At 14:30, the robust

Table 2. The standard and robust TE processes in detail

	Condition of the network	Optimization expected values	Condition of the network	Optimization expected values	Condition of the network	Optimization expected values
Run	max util. cost	max util. cost	max util. cost	max util. cost	max util. cost	max util. cost
12:00, std	71.55% 1.68e7	50.51% 1.54e7	**50.51% 1.54e7**	53.05% 1.52e7		
12:00, rob	71.55% 1.68e7	65.58% 1.57e7	**57.89% 1.55e7**	57.89% 1.55e7		
14:30, std	84.72% 1.86e7	49.85% 1.68e7	**49.85% 1.68e7**	49.85% 1.68e7		
14:30, rob	84.72% 1.86e7	66.58% 1.68e7	75.26% 1.76e7	51.55% 1.67e7	72.61% 1.82e7	51.55% 1.71e7
	51.55% 1.68e7	51.55% 1.68e7				
15:00, std	83.24% 1.79e7	52.35% 1.61e7	**52.35% 1.58e7**	52.35% 1.58e7		
15:00, rob	83.24% 1.79e7	52.35% 1.61e7	**52.35% 1.61e7**	52.35% 1.61e7		
15:30, std	81.88% 1.75e7	49.27% 1.58e7	**49.27% 1.58e7**	49.27% 1.58e7		
15:30, rob	81.88% 1.75e7	49.27% 1.59e7	**51.18% 1.58e7**	51.18% 1.58e7		
16:00, std	70.27% 1.61e7	49.64% 1.53e7	72.69% 1.67e7	48.05% 1.56e7	**48.05% 1.57e7**	48.05% 1.57e7
16:00, rob	70.27% 1.61e7	48.05% 1.55e7	**48.05% 1.55e7**	48.05% 1.55e7		
16:30, std	78.40% 1.80e7	52.54% 1.55e7	76.99% 1.72e7	48.96% 1.56e7	**48.96% 1.53e7**	**48.96% 1.53e7**
16:30, rob	78.40% 1.80e7	52.54% 1.57e7	**52.54% 1.58e7**	52.54% 1.57e7		
17:00, std	73.96% 1.60e7	49.18% 1.50e7	74.74% 1.63e7	49.18% 1.51e7	**49.18% 1.50e7**	49.18% 1.48e7
17:00, rob	73.96% 1.60e7	49.18% 1.50e7	70.87% 1.64e7	66.30% 1.56e7	**66.30% 1.53e7**	49.18% 1.51e7
20:00, std	76.43% 1.50e7	46.62% 1.38e7	**46.62% 1.39e7**	46.62% 1.38e7		
20:00, rob	76.43% 1.50e7	53.50% 1.39e7	**53.50% 1.37e7**	53.50% 1.37e7		

TE requires 4 optimization runs, wheres the standard one requires 2 runs. On the contrary, at 16:00 and 16:30. the standard TEs require one more optimization run than the robust ones do. Thus, we can not say, the robust TE beats the standard one in terms of required number of optimization runs. We can also observe that with the standard TE the final condition of the network is better at all times except at 16:00 and 20:00. This is due to the fact that the standard TE focuses on a single traffic matrix to optimize where the robust optimizes multiple matrices simultaneously.

In Figure 4, we compare the robustness of the TE approaches. The points in the figure give the maximum utilization rates which are obtained with the resulting weight setting and traffic matrices having been formed during each run. For example, during the robust TE process for 12:00, three different traffic matrices have been formed and the maximum utilizations with the final weight setting against these traffic matrices are 57.89%, 65.58% and 50.51%. We can observe that the worst three performance is obtained with the standard approach.

In the last part of our experiments, we are interested in the performances of the TE processes when the network conditions are more strict. To do that, we have carried out an additional modification in the topology; the capacities of 32 links are divided further by 5. Both standard and robust TE processes are run for the NetFlow data dumped at 15:30. The details of these runs are given in Table 3. Please note, that the only the maximum utilization rates are given in order to save place. In these runs, the standard TE process does not stop by the given conditions. We forced it to stop after 15^{th} optimization. However, the robust TE process stops after the third optimization, since the robust weight setting it provides at the second optimization works well for the next traffic matrix. The third weight setting can not provide a dramatic increase in the state of the network. Thus, the robust TE outperforms the standard one when the network resources are more strict.

Additionally, we observe that hot-potato routing can even cause the network to obtain a performance which is worse than the one before the optimization step. The network starts with a maximum utilization rate of 82.70%, but after the implementation of the optimal weights the utilization rate increases to 91.07%.

8 Conclusion

In this paper, we investigated the negative effects of hot-potato routing on the IGP link weight optimizer. The decision process carried out in BGP may cause the optimized weights to perform worse than the expected, since hot-potato routing may change the traffic matrix of the network with the updated weight setting such that this weight setting does not work well anymore for the new traffic matrix. In this paper, with a worst-case example we showed that the interaction of the optimizer with BGP may cause a convergence failure in the standard TE process. Some case studies were also carried out in this paper to observe the possible negative effects of the IGP-WO and BGP inter-

Table 3. The standard and robust TE processes for 15:30 with modified topology

Run	Network condition	Opt. exp. values	Network condition	Opt. exp. values	Network condition	Opt. exp. values	Network condition	Opt. exp. values	Network condition	Opt. exp. values
std	82.70%	67.31%	91.07%	60.36%	93.12%	51.21%	85.31%	60.08%	87.90%	49.96%
	89.69%	61.50%	87.32%	49.46%	90.93%	60.74%	88.00%	52.31%	90.49%	78.96
	89.60%	52.31%	90.67%	78.96%	89.23%	49.46%	90.67%	78.96%	89.24%	49.46%
rob	82.70%	67.31%	91.07%	60.36%	50.96%	50.96%				

action in a real network. In the case studies, we observed that due to hot-potato routing the network performs worse than the starting point with the optimized weights. In this study, a TE approach based on the robust optimization of the link weights against multiple traffic matrices was proposed to smoothen the negative effect of hot-potato routing. In the experiments, we have seen that the robust TE process is especially more effective when the network resources are more strict, i.e. it stops much earlier than the standard one. Additionally, the robust TE provides solutions which are not necessarily best for the last traffic matrices but good for all of the previously generated matrices. As a future research, we are interested in alternative approaches to overcome the negative interaction of BGP with IGP-WO.

Acknowledgments

This work was partially supported by the Walloon Region (Belgium) in the framework of the WIST program (TOTEM project) and by the E-NEXT European Network of Excellence. We are grateful to Nicolas Simar (DANTE, UK) for the information given about GÉANT, to Steve Uhlig (UCL, Belgium) for aggregating the Netflow traces, and to Olivier Bonaventure for his valuable comments.

References

[1] S. Agarwal, A. Nucci, and S. Bhattacharyya. Measuring the shared fate of IGP engineering and interdomain traffic. In *Proceedings of ICNP 2005*, 2005.

[2] A. Bley, M. Grötchel, and R. Wessäly. Design of broadband virtual private networks: Model and heuristics for the B-Win. In *Proceedings of DIMACS Workshop on Robust Communication Networks and Survivability*, volume 53 of *AMS-DIMACS Series*, pages 1–16, 1998.

[3] R. Callon. Use of OSI IS-IS for routing in TCP/IP and dual environments. RFC 1195, December 1990.

[4] C-BGP - an efficient BGP simulator. `http://cbgp.info.ucl.ac.be/`, 2005.

[5] Bgp best path selection algorithm. `http://www.cisco.com/warp/public/459/25.shtml`.

[6] M. Ericsson, M. G. C. Resende, and P. M. Pardalos. A genetic algorithm for the weight setting problem in OSPF routing. *Journal of Combinatorial Optimization*, 6(3):299–333, 2002.

[7] B. Fortz, J. Rexford, and M. Thorup. Traffic engineering with traditional ip routing protocols. *IEEE Communications Magazine*, 40(10):118–124, 2002.

[8] 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.

[9] B. Fortz and M. Thorup. Increasing internet capacity using local search. *Computational Optimization and Applications*, 29:13–48, 2004.

[10] Foundry enterprise configuration and management guide. `http://www.foundrynet.com/services/documentation/ecmg/BGP4.html\#17143`.

[11] The GÉANT network. `http://www.geant.net`.

[12] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publisher, 1997.

[13] K. Holmberg and D. Yuan. Optimization of Internet Protocol network design and routing. *Networks*, 43(1):39–53, 2004.

[14] Interior gateway protocol weight optimization tool. `http://www.poms.ucl.ac.be/totem/`.

[15] G. Leduc, H. Abrahamsson, S. Balon, S. Bessler, M. D'Arienzo, O. Delcourt, J. Domingo-Pascual, S. Cerav-Erbas, I. Gojmerac, X. Masip, A. Pescapé, B. Quoitin, S. P. Romano, E. Salvadori, F. Skivée, H. T. Tran, S. Uhlig, and H. Ümit. An open source traffic engineering toolbox. *Computer Communications*, 2005.

[16] J. Moy. OSPF Version 2. RFC 2328, April 1998.

[17] M. Network. Mrt: multi-theraded routing toolkit. `http://www.mrtd.net/`.

[18] B. Quoitin and S. Tandel. A BGP solver for hot-potato routing sensitivity analysis. In *Proceedings of EUNICE 2005*, 2005.

[19] B. Quoitin and S. Tandel. C-BGP user's guide, 2005.

[20] Y. Rekther, T. Li, and S. Hares. A Border Gateway Routing Protocol 4 (BGP-4). Internet draft, `http://www.ietf.org/internet-drafts/draft-ietf-idr-bgp4-26.txt`, October 2004.

[21] The sprint network. `http://www.sprintworld.com`.

[22] R. Teixeira, A. Shaikh, T. Griffin, and J. Rexford. Dynamics of hot-potato routing in IP networks. In *Proceedings of SIGMETRICS/Perfomance'04*, 2004.

[23] R. Teixeira, A. Shaikh, T. Griffin, and G. M. Voelker. Network sensitivity to hot-potato disruptions. In *Proceedings of ACM SIGCOMM'04*, 2004.

[24] Totem toolbox. `http://totem.run.montefiore.ulg.ac.be`, 2005.

[25] S. Uhlig. On the sensitivity of transit ASes to internal failures. In *Proc. of the 5th IEEE International Workshop on IP Operations and Management (IPOM 2005), LNCS 3751*, pages 142–151, 2005.