Taming traffic dynamics : analysis and improvements

Pedro Casas Hernandez, Federico Larroca, Jean Louis Rougier, Sandrine Vaton

Taming Traffic Dynamics: Analysis and Improvements

Pedro Casas[a,b], Federico Larroca[c,*], Jean-Louis Rougier[c], Sandrine Vaton[a]

[a] *Télécom Bretagne. Technopôle Brest-Iroise, CS 83818, 29238 Brest Cedex 3, France*
[b] *Facultad de Ingeniería, Universidad de la República. Julio Herrera y Reissig 565, C.P. 11.300, Montevideo, Uruguay*
[c] *Télécom ParisTech. 46 rue Barrault, F-75634 Paris Cedex 13, Paris, France*

Abstract

Internet traffic is highly dynamic and difficult to predict in current network scenarios, which enormously complicates network management and resources optimization. To address this uncertainty in a robust and efficient way, two almost antagonist Traffic Engineering (TE) techniques have been proposed in the last years: Robust Routing and Dynamic Load-Balancing. Robust Routing (RR) copes with traffic uncertainty in an off-line preemptive fashion, computing a single static routing configuration that is optimized for traffic variations within some predefined uncertainty set. On the other hand, Dynamic Load-Balancing (DLB) balances traffic among multiple paths in an on-line reactive fashion, adapting to traffic variations in order to optimize a certain congestion function. In this article we present the first comparative study between these two alternative methods. We are particularly interested in the performance loss of RR with respect to DLB, and on the response of DLB when faced with abrupt changes. This study brings insight into several RR and DLB algorithms, evaluating their virtues and shortcomings, which allows us to introduce new mechanisms that improve previous proposals.

Keywords: Traffic Uncertainty, Traffic Management, Robust Optimization, Robust Routing, Dynamic Load-Balancing

1. Introduction

As network services and Internet applications evolve, network traffic is becoming increasingly complex and dynamic. The convergence of data, telephony and television services on an all-IP network directly translates into a much higher variability and complexity of the traffic injected into the network. To make matters worse, the presence of unexpected events such as network equipment failures, large-volume network attacks, flash crowd occurrences and even external routing modifications induces large uncertainty in traffic patterns. Moreover, current evolution and deployment-rate of broadband access technologies (e.g. Fiber To The Home) only aggravates this uncertainty.

But these are not the only problems network operators are confronted with. The ever-increasing access rates available for end-users we just mentioned is such that the assumption of infinitely provisioned core links could soon become obsolete. In fact, recent Internet traffic studies from major network technology vendors like Cisco Systems forecast the advent of the Exabyte era [1, 2], a massive increase in network traffic driven by high-definition video. In this context, simply upgrading link capacities may no longer be an economically viable solution. Moreover, even if overdimensioning would be possible, its environmental impact is not negligible. For instance, the Information and Communication Technology sector alone is responsible for around 2% of the man-made CO_2, a similar figure to that of the airline industry, but with higher increasing perspectives [3]. An efficient and responsible usage of the resources is then essential[1].

In the light of this traffic scenario, we study the problem of intradomain Traffic Engineering (TE) under traffic uncertainty. This uncertainty is assumed to be an exogenous traffic modification, meaning that traffic variations are not produced within the domain for which routing is optimized but are due to external and difficult to predict events. More in particular, we are interested in two almost antagonist approaches that have emerged in the recent years to cope with both the increasing traffic dynamism and the need for cost-effective solutions: Robust Routing (RR) [6, 7, 8] and Dynamic Load-Balancing (DLB) [9, 10, 11].

In RR, traffic uncertainty is taken into account directly within the routing optimization, computing a single routing configuration for all traffic demands within some *uncertainty set* where traffic is assumed to vary. This uncertainty set can be defined in different ways, depending on

*Corresponding author. Telephone: +33 (0)1 45 81 75 52 - Fax: +33 (0)1 45 89 79 06

Email addresses: pedro.casas@telecom-bretagne.eu (Pedro Casas), federico.larroca@telecom-paristech.fr (Federico Larroca), rougier@telecom-paristech.fr (Jean-Louis Rougier), sandrine.vaton@telecom-bretagne.eu (Sandrine Vaton)

[1] To learn more about this emerging discipline, the interested reader should consult works related to so-called "green networking" [4].

the available information: largest values of links load previously seen, a set of previously observed traffic demands (previous day, same day of the previous week), etc. The criterion to search for this unique routing configuration is generally to minimize the maximum link utilization (i.e. the utilization of the most loaded link in the network) for all traffic demands of the corresponding uncertainty set. While this routing configuration is not optimal for any single traffic demand within the set, it minimizes the worst case performance over the whole set.

DLB copes with traffic uncertainty and variability by splitting traffic among multiple paths in real-time. In this dynamic scheme, each origin-destination (OD) pair of nodes within the network is connected by several a priori configured paths, and the problem is simply how to distribute traffic among these paths in order to optimize a certain function. DLB is generally defined in terms of a link-congestion function, where the portions of traffic are adjusted in order to minimize the total network congestion. Ideally, the traffic distribution is set so that at every instant the objective function is optimized.

Those who promote DLB highlight among others the fact that it is the most resource-efficient possible scheme, and that given the configured paths it supports every possible traffic demand, all of this in an automated and decentralized fashion. In practice, the "always-optimized" characteristic we mentioned above is achieved by means of a distributed algorithm periodically executed by every ingress router based on feedback from the network. It is precisely this last characteristic that constitutes the most challenging aspect of DLB. In fact, the deployment of DLB has been, to say the least, limited. Two particular problems arise in DLB: convergence to the optimum is not always guaranteed, and convergence speed might be over-killing under large and abrupt changes in traffic demands. Network operators are reluctant to use dynamic mechanisms mainly because they are afraid of a possible oscillatory behavior of the algorithm used by each OD pair to adjust load-balancing. As the early experiences in ArpaNet has proved [12], these concerns are not without reason. (In particular, before July 1987, the links' metric was defined as the packet delay averaged over a 10 s period. Although this adaptive routing scheme worked correctly under light or moderate loads, it generated oscillations under relatively heavy loads. This resulted in substituting the links' metric by a fixed value as we use it today, sacrificing optimality for stability.) Indeed, for these adaptive and distributed algorithms, a trade-off between adaptability (convergence speed) and stability must be found, which may be particularly difficult in situations where abrupt traffic changes occur.

Those who advocate the use of RR claim that there is actually no need to implement supposedly complicated and possibly oscillatory dynamic routing mechanisms, and that the incurred performance loss for using a single routing configuration is negligible when compared with the increase in complexity. RR provides a stable routing configuration for all the traffic demands within the uncertainty set, avoiding possible oscillations and convergence issues. However, RR presents some conception problems and serious shortcomings in its current state which we highlight and try to ease in this work. The first drawback of current RR is related to the objective function it intends to minimize. Optimization under uncertainty is generally more complex than classical optimization, which forces the use of simpler optimization criteria such as maximum link utilization (MLU). The MLU is not the most suitable network-wide optimization criterion; setting the focus too strictly on MLU often leads to worse distribution of traffic, adversely affecting the mean network load and thus the total network end-to-end delay, an important QoS indicator. It is easy to see that the minimization of the MLU in a network topology with heterogeneous link capacities may lead to poor results as regards global network performance. The second drawback of RR we identify is its inherent dependence on the definition of the uncertainty set of traffic demands: the uncertainty set has to be sufficiently "large" to allow traffic flexibility and to provide performance guarantees, but should not be excessively "large" to avoid wasting network resources. Thus, considering a unique RR configuration to address both traffic in normal operation and unexpected traffic variations is an inefficient strategy, as a single routing configuration cannot be suitable for both situations.

1.1. Contributions of this article

This article presents a fair and comprehensive comparative analysis between RR and DLB mechanisms. The analysis is comprehensive as it evaluates the performance of both mechanisms based on different performance indicators and considering normal operation as well as unpredicted traffic events. We believe our comparison is fair because it considers the particular characteristics of each mechanism under the same network and traffic conditions. To date and to the best of our knowledge this is the first work that conducts such a comparative evaluation, necessary indeed not only from a research point of view but also for network operators who seek cost-effective and robust solutions to face future network scenarios. Based on this comparative analysis we develop and evaluate new variants of RR and DLB mechanisms, improving some of the shortcomings found in both static and dynamic approaches.

Regarding the RR approach, we will introduce some modifications that strive to alleviate the two problems identified in current proposals. We will first study which is the best objective function to minimize, and propose the mean link utilization instead of the MLU. The mean link utilization provides a better image of network-wide performance, as it does not depend on the particular load or capacity of each single link in the network but on the average value. However, a direct minimization of the mean link utilization does not assure a bounded MLU, which is not practical from an operational point of view. Thus, we minimize the mean link utilization while bounding the MLU

by a certain utilization threshold a priori defined. This adds a new, and maybe difficult to set, constraint to the problem, namely how to define this utilization threshold. We further improve our proposal by providing a multiple objective optimization criterion, where both the MLU and the mean link utilization are minimized simultaneously. We evaluate the improvements of our proposals from a QoS perspective, using the mean path end-to-end queuing delay as a measure of global performance.

The second problem we address in RR is the trade-off between routing performance and routing reliability. In [13] we have recently proposed a solution to manage this trade-off, known as Reactive Robust Routing (RRR). Basically, RRR consists of constructing a RR configuration for expected traffic in nominal operation, adapting this nominal routing configuration after the detection and localization of a large and long-lived traffic modification. RRR provides good performance for both nominal operation and unexpected traffic, but it is difficult to deploy in a real implementation, because of the routing reconfiguration step. Reconfiguring the routing of an entire Autonomous System is a nontrivial task. In this article we modify the RRR approach, using a preemptive Load Balancing algorithm to balance traffic among pre-established paths after the localization of a large volume traffic modification (preemptive in the sense of preventing a situation from occurring).

In what respects DLB, we evaluate the use of so-called *no-regret* algorithms as the distributed optimization algorithm used by ingress routers to adapt load-balancing. The authors of a recent paper [14] proved that if all OD pairs use algorithms of this kind, convergence to the optimum is guaranteed. Special attention will be paid on the behavior of the algorithm when faced with abrupt and unexpected changes in the traffic demands. We shall introduce simple, and yet effective, modifications to the algorithm to assure a fast convergence to the new optimum in this case.

As we shall see in the following subsection, several previously proposed DLB algorithms strive to minimize the MLU by means of a greedy algorithm in the paths utilization (i.e. each ingress router increases the amount of traffic sent along the path with the smallest utilization). As proved in a recent paper [15], convergence to the optimum for such algorithms may not be guaranteed, in the sense that they may converge to a situation in which the MLU is not minimized. In fact, we shall present an example in which the difference with the optimum of the MLU is non-negligible. In this work we will present an alternative path cost function, so that greedy algorithms that use it do converge to the optimum equilibrium.

1.2. Related Work

There is a large literature on routing optimization with uncertain traffic demands. Thus, here we shall only mention a few papers, and do not expect our list to be exhaustive.

Traditional algorithms rely on a single or a small group of expected traffic demands to compute optimal and reliable routing configurations. An extreme case is presented in [16], where routing is optimized for a single estimated traffic demand and is then applied for daily routing. Traffic uncertainty is characterized by multiple traffic demands in [17] (set of traffic demands from previous day, same day of previous week, etc.), where different mechanisms to find optimal routes for the set are presented. As discussed for instance in [18], this perspective is no longer suitable for current and future dynamic scenarios. These approaches require a "leap of faith" to perform well, mainly because they assume that traffic patterns do not change that much over time. However, even a relatively small difference between the "real" traffic demand and the one used for the routing optimization may lead to an important performance degradation. Such a difference may arise in the event of unexpected traffic variations (which are more common nowadays), or even be due to an error in the traffic estimation.

A different approach has emerged in the recent years to cope with the increasing traffic dynamism and the need for cost-effective solutions, Dynamic Load-Balancing (DLB) [9, 10, 11, 19]. In DLB, traffic is split among a priori established paths in order to avoid network congestion. The two most well-known proposals in this area are MATE and TeXCP. In MATE [9], a convex link congestion function is defined, which depends on the link capacity and the link load. The objective is to minimize the total network congestion, for which a simple gradient descent method is proposed. In [19], we propose to use a link congestion function based on measurements of the queueing size, which results in better global performance from a QoS perspective. TeXCP [10] proposes a somewhat simpler objective: in order to minimize the MLU, they minimize the biggest utilization each traffic demand obtains in its paths. Another DLB scheme which has the same objective but a relatively different mechanism is REPLEX [11].

The last category of algorithms consists of Robust Routing techniques [6, 7, 8, 20, 21]. The objective in RR is to find a unique static routing configuration that fulfills a certain criterion for a broad set of traffic demands, generally the one that minimizes the maximum link utilization over the whole set of demands. In [6], authors capture traffic variations by introducing a polyhedral set of demands, which allows for easier and faster linear optimization. This robust technique is applied in [20] to compute a robust MPLS routing configuration without depending on traffic demand estimation, and corresponding methods for robust OSPF optimization are discussed. Oblivious Routing [7] also defines linear algorithms to optimize worst-case MLU for different sizes of traffic uncertainty sets. The author of [21] analyzes the use of robust routing through a combination of traffic estimation techniques and its corresponding estimation error bounds, in order to shrink the set of traffic demands. In [8] authors introduce COPE, a RR mechanism that optimizes routing for

predicted demands and bounds worst-case MLU to ensure acceptable efficiency under unexpected traffic events. The idea behind COPE is similar to ours, in the sense that it strives to alleviate performance degradation due to unforeseen traffic modifications. Nevertheless, it proposes a single routing configuration to handle expected as well as large and abrupt traffic variations, which is clearly not the best solution.

The same paper [8] presents, to the best of our knowledge, the only previous comparative study between RR and DLB. The authors of the paper compare the performance of COPE with a dynamic approach which they claim models the behavior of mechanisms such as MATE and TeXCP. Given a time series traffic demands, this dynamic approach consists of computing an optimal routing for each traffic demand i and evaluate its performance with the following traffic demand $i+1$. There are two important shortcomings of this DLB simulation. Firstly, adaptation in DLB is iterative and never instantaneous. Secondly, in all DLB mechanisms paths are set a priori and remain unchanged during operation. This is not the case in their dynamic approach, where each new routing optimization may change not only traffic portions but paths themselves. For these reasons, we believe that the comparison provided in [8] is biased against dynamic schemes.

The remainder of this article is organized as follows. In Sec. 2 we introduce the network model and notation, while Sec. 3 and 4 introduce a preliminary version of the RR and DLB mechanisms. Some first results are discussed in Sec. 5. Section 6 presents new variants to the former mechanisms which alleviate the shortcomings detected. The evaluation of the complete set of algorithms under different traffic scenarios is conducted in Sec. 7. We finally draw conclusions of this comparative analysis in Sec. 8.

2. Network Model and Performance Indicators

Let us begin by introducing the notation used in this article. The network topology is defined by n nodes and a set $L = \{l_1, \ldots, l_q\}$ of q links, each with a corresponding capacity c_i, $i = 1, \ldots, q$. The Traffic Matrix (TM) $X = \{x_{i,j}\}$ denotes the traffic demand (expressed in, for example, Mbps) between every origin node i and every destination node j ($i \neq j$) of the network; we shall note each of these origin-destination pairs as OD pairs, and each origin-destination traffic demand $x_{i,j}$ as OD flows. In practice, each traffic demand is measured every T minutes (usually 5' or 10' [5]), and its value simply corresponds to the cumulative number of bytes observed between two consecutive measurements, divided by the polling time T. Let $X = \{x_k\}$ be the vector representation of the TM, where we have reordered OD flows by index $k = 1, \ldots, m$ ($m = n.(n-1)$). Let $N = \{\mathrm{OD}_1, \ldots, \mathrm{OD}_m\}$ be the set of m OD pairs. We consider a multi-path network topology, where each OD flow x_k can be arbitrarily split among a set of p_k origin-destinations paths P_k. In this sense, we shall call r_p^k the portion of traffic flow x_k sent along path $p \in P_k$, where $0 \leqslant r_p^k \leqslant 1$ and $\sum_{p \in P_k} r_p^k = 1$.

Let λ_l^p be an indicator variable that takes value 1 if path p traverses link l and 0 otherwise, and $Y = \{\rho_1, \ldots, \rho_q\}$ a vector representation of links traffic load. Then X and Y are related through the routing matrix R, a $q \times m$ matrix $R = \{r_l^k\}$ where $r_l^k = \sum_{p \in P_k} \lambda_l^p \,.\, r_p^k$. The variable r_l^k indicates the fraction of OD flow x_k routed along link l; this results in the following relation:

$$Y = R \,.\, X \tag{1}$$

Given X, the multi-path routing optimization problem consists in choosing the set of paths P_k for each OD pair k and computing the routing matrix R, in order to optimize a certain objective function $g(X, R)$. A simplified version of this problem is the load-balancing optimization problem which, *given* a set of paths, calculates R. In this work we shall consider different performance indicators, which result in different objective functions.

A very important link-level performance indicator is the link utilization $u_l = \rho_l / c_l$; a value of u_l close to one indicates that the link is operating near its capacity. Network operators usually prefer to keep links utilization relatively low in order to support sudden traffic increases and link/node failures. A network-wide performance indicator is the maximum link utilization $u_{\max}$:

$$u_{\max}(X, R) = \max_{l \in L} \{u_l\} \tag{2}$$

The maximum link utilization constitutes by far the most popular TE objective function. However, its minimization presents a clear drawback: setting the focus too strictly on the most utilized link often leads to a worse distributions of traffic, adversely affecting the overall performance in the network. In this sense we will consider the mean link utilization u_{mean} as another possible objective function:

$$u_{\mathrm{mean}}(X, R) = \frac{1}{q} \sum_{l \in L} u_l \tag{3}$$

Minimizing u_{mean} may provide better network-wide performance, as long as the maximum link utilization remains bounded; we will further discuss this issue in Sec. 6.

The last performance indicator we shall consider in this work is the queuing delay (i.e. the time that spans between the moment a byte enters the router and leaves it). This choice is justified by two aspects. Firstly, its algebra is relatively simple, in the sense that the total delay of a path is the addition of the delay at each link. Secondly, it is a very versatile indicator. A big queueing delay means more delay and jitter for streaming traffic. Moreover, a link with an important queueing delay is traversed by several bottlenecked flows, meaning that elastic traffic may obtain better throughput in other, less loaded, links. The mean queueing delay may be regarded then as a numerical value of the congestion on the link.

Assume then that the queuing delay on link l is given by the function $d_l(\rho_l)$. Given this function we can compute the queuing delay of path p as $d_p = \sum_{l \in p} d_l(\rho_l)$. As a measure of the network-wide performance, we consider the expected end-to-end (e2e) queuing delay d_{mean} defined as:

$$d_{\text{mean}}(X, R) = \sum_{k \in N} \sum_{p \in P_k} \left(r_p^k \,.\, x_k\right) d_p = \sum_{l \in L} \rho_l \,.\, d_l(\rho_l) \quad (4)$$

That is to say, d_{mean} is a weighted mean e2e queuing delay, where the weight for each path is how much traffic is sent along it ($r_p^k \,.\, x_k$), or in terms of links, the weight for each link is how much traffic is traversing it (ρ_l). We prefer a weighted mean queuing delay to a simple total delay because it reflects more precisely performance as perceived by traffic. Two situations where the total delay is the same, but in one of them most of the traffic is traversing heavily delayed links should not be considered as equivalent. Note that, by Little's law, the value $f_l(\rho_l) = \rho_l \,.\, d_l(\rho_l)$ is proportional to the volume of data in the queue of link l. We will then use this last value as the addend in the last sum in (4), since it is easier to measure than the queuing delay. Finally, note that, differently to $u_{\max}$, a large mean e2e queuing delay translates into bad performance for the majority of the traffic and not only for the traffic that traverses a particularly loaded link.

Based on these definitions we will introduce the different optimization algorithms that strive to minimize some of these performance indicators, considering either the RR or DLB approach.

3. Stable Robust Routing

Finding a multi-path routing configuration minimizing $u_{\max}$ is an instance of the classical multi-commodity flow problem which can be formulated as a linear program [22]. For a single known traffic matrix X, the problem can be easily solved by linear programming techniques [23]. However, as we have previously discussed, traffic demands are uncertain and difficult to predict, and all we can expect is to find them within some bounded uncertainty set.

In a robust perspective of the multi-path routing optimization problem, demand uncertainty is taken into account within the routing optimization, computing a single routing configuration for all demands within some uncertainty set. In this work we consider a polyhedral uncertainty set $\mathbb{X}$, more precisely a *polytope* as in [6], based on the intersection of several half-spaces that result from linear constraints imposed to traffic demand.

As an example, let us define an uncertainty set $\mathbb{X}$ based on a given routing matrix R_{o} and the peak-hour links traffic load Y^{peak} obtained with this routing matrix:

$$\mathbb{X} \;=\; \left\{X \in \mathbb{R}^m, R_{\text{o}}.X \leqslant Y^{\text{peak}},\; X \geqslant 0\right\}$$

Observe that this definition of the uncertainty set has a major advantage: routing optimization can be performed from easily available links traffic load Y without even knowing the actual value of the traffic demand X. Figure 1 depicts the obtained uncertainty set, based on the convex intersection of q half-spaces of the form $\mathbf{r}_{\text{o}}^i \cdot X \leqslant \rho_i^{\text{peak}}$, $\forall i \in L$, where $\mathbf{r}_{\text{o}}^i$ stands for the i-th row of the routing matrix R_{o}.

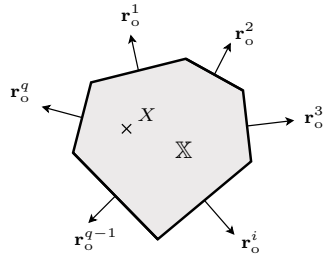


Figure 1: The uncertainty set $\mathbb{X}$ as a polytope.

	minimize	$u_{\max}$		
	subject to:			
$\sum_{k \in N} \sum_{p \in P_k} \lambda_l^p \,.\, r_p^k \,.\, x(k)$	$\leqslant$	$u_{\max}.c_l$	$\forall\, l \in L,\ \forall\, X \in \mathbb{X}$	
$\sum_{p \in P(k)} r_p^k$	$=$	1	$\forall\, k \in N$	
r_p^k	$\geqslant$	0	$\forall\, p \in P_k,\ \forall\, k \in N$	
$u_{\max}$	$\leqslant$	1		

Table 1: The Robust Routing Optimization Problem (RROP)

The traditional *Robust Routing Optimization Problem* (RROP) defined in Table 1 consists of minimizing the maximum link utilization $u_{\max}$, considering all demands within $\mathbb{X}$. The solution to the problem is twofold: on the one hand, a routing configuration R_{robust}, and on the other hand, a worst-case performance threshold $u_{\max}^{\text{robust}}$:

$$\begin{aligned} R_{\text{robust}} &= \operatorname*{argmin}_R \max_{X \in \mathbb{X}} u_{\max}(X, R) \\ u_{\max}^{\text{robust}} &= \max_{X \in \mathbb{X}} u_{\max}(X, R_{\text{robust}}) \end{aligned}$$

Given a suitable definition of the uncertainty set, the obtained robust routing configuration R_{robust} is applied during long periods of time; in this sense, we refer to Robust Routing as *Stable Robust Routing* (SRR). The authors of [6] have shown that the RROP can be efficiently solved by linear programming techniques, applying a combined columns and constraints generation method. This method iteratively solves the problem, progressively adding new constraints and new columns to the problem.

The new constraints are the extreme points of the uncertainty set $\mathbb{X}$, and the new columns represent new paths added to reduce the objective function value. Only extreme points of $\mathbb{X}$ are added as new constraints, as it is easy to see that every traffic demand $X \in \mathbb{X}$ can be expressed as a linear combination of these extreme demands. Regarding new added paths, the algorithm in [6] may not be the best choice from a practical point of view since the number of paths for each OD pair is not a priori restricted

and the characteristics of added paths are not controlled. For example, it would be interesting to have disjoint paths to route traffic from each single OD pair, improving resilience. For this reason we modify the algorithm to select new paths, both limiting the maximum number of paths in P_k and taking as new candidates the shortest paths with respect to link weights w_l^i:

$$w_l^i = \frac{1}{\epsilon + \left(1 - {r_l^k}^i\right)} \tag{5}$$

where ${r_l^k}^i$ corresponds to the fraction of traffic flow x_k that traverses link l after iteration i and ϵ is a small constant that avoids numerical problems. If OD pair k uses a single path p at iteration i, ${r_l^k}^i = 1$ for every link $l \in p$, and so this path is removed from the graph where new shortest paths are computed ($w_l \to \infty$, $\forall l \in p$). While this may result in a sub-optimal performance, it allows a real and practical implementation. In case there are no disjoint paths for OD pair k, we use the column constraint generation method used in [6] to add new paths for OD pair k.

4. Dynamic Load Balancing

4.1. Routing Games and Wardrop Equilibrium

As mentioned before, the objective in DLB is to minimize a certain objective function $g(X, R)$ in a distributed fashion (i.e. without relying on any centralized entity). Algorithms that achieve this are typically greedy, which present the desirable property of requiring minimum coordination among border routers. In this kind of mechanisms, a path cost function ϕ_p is defined, and each OD pair greedily minimizes the cost it obtains from each of its paths. This context constitutes an ideal case study for game theory, and is known as *Routing Game* in its terminology [24, 25].

Since each OD pair may arbitrarily balance traffic among its paths, we will assume that OD pairs are constituted of infinitely many agents. These agents control an infinitesimal amount of traffic, and decide along which path to send their traffic. In this context r_p^k represents then the fraction of agents of OD pair k that have p as their choice. If each of these agents acts selfishly, then the system will be at equilibrium when no agent may decrease its cost by unilaterally changing its path decision. This situation constitutes what is known as a *Wardrop Equilibrium* (WE) [26], which is formally defined as follows:

Definition 1. The paths vector $\{r_p^k\}_{k\in N, p\in P_k}$ is a Wardrop Equilibrium if for each OD pair $k \in N$ and for each couple of paths $p, q \in P_k$ with $r_p^k > 0$ it holds that $\phi_p \leq \phi_q$.

Intuitively speaking, a WE is a situation where each OD pair uses only those paths with minimum cost (for the given OD pair). Anyway, the path cost ϕ_p is in turn defined in terms of a certain nonnegative, nondecreasing and continuous link cost function $\phi_l(\rho_l)$. There are roughly two kinds of games depending on the definition of ϕ_p. A *Congestion Routing Game* defines the path cost as $\phi_p = \sum_{l\in p} \phi_l(\rho_l)$. On the other hand, a *Bottleneck Routing Game* defines $\phi_p = \max_{l\in p} \phi_l(\rho_l)$.

Much effort has been put into characterizing the resulting equilibrium of these games. In this sense, a certain social cost function is defined, which measures the dissatisfaction of the OD pairs as a whole (i.e. an optimum paths vector is one that minimizes this function), and the objective is to quantify the difference between the optimum and the resulting WE. In the case of a congestion game, the typical social cost function is the same as in (4) (i.e. $\sum_{l\in L} \rho_l d_l(\rho_l) := \sum_{l\in L} f_l(\rho_l)$), whereas for a bottleneck game the social cost is usually the maximum $\phi_l(\rho_l)$ over all links (i.e. $\max_{l\in L} \phi_l(\rho_l)$).

It may be proved that the WE of a congestion game coincides with the unique minimum of the so-called potential function $\Phi(R) = \sum_{l\in L} \int_0^{\rho_l} \phi_l(x)dx$ [24]. This means that if $f_l(\rho_l)$ is continuous differentiable, non-decreasing and convex, the WE of a congestion game with $\phi_l(\rho_l) = f_l'(\rho_l)$ is socially optimum. In this sense, to minimize d_{mean} through DLB, we will play a Congestion Routing Game with a link cost equal to the derivative of the link mean queue size. In the sequel we shall note this game as MinDG (Minimum Delay Game).

On the other hand, characterization of the WE of a bottleneck game is somewhat more complicated. In fact, it is relatively easy to see that in this case the WE is not even unique. Moreover, and rather unfortunately, it has been proved in [15] that even if there always exist at least one WE that is socially optimum, nothing may be guaranteed about the rest (if any). However, the same paper proved that every WE that fulfills the so-called *efficiency condition* is optimum, where this condition is defined as follows:

Definition 2. Let $B(p)$ denote the number of network bottlenecks over p; that is to say $B(p) = \left|\left\{l \in p : \phi_l(\rho_l) = \max_{m\in L, \rho_m>0} \{\phi_m(\rho_m)\}\right\}\right|$. Then, a WE is said to satisfy the efficiency condition if all OD pairs route their traffic along paths with a minimum number of network bottlenecks; i.e. for all $k \in N$ and $p, q \in P_k$ with $r_p^k > 0$ it holds that $B(p) \leq B(q)$.

This result, which is relatively new, was not applied in the design of neither TeXCP or REPLEX, both of which strive to minimize the maximum link utilization by means of a greedy algorithm in the path utilization (i.e. a bottleneck game with $\phi_p = \max_{l\in p} u_l$). It could then be the case that these algorithms converge to a sub-optimal WE. Possible consequences on the obtained performance of ignoring this result will be further discussed later in the article. In any case, we shall note this game as MinUG (Minimum Utilization Game).

4.2. No-Regret Algorithms

We will now briefly discuss how, given the path cost function ϕ_p, the WE may be achieved for both routing games. In a recent article [14], the authors proved that if all OD pairs use *no-regret* algorithms, the global behavior will approach the WE. To be more precise, for a given TM X, and for *most* time steps, the instantaneous paths vector $\{r_p^k\}_{k\in N, p\in P_k}$ is very close to the WE, and this difference vanishes with time.

This result is very general, in the sense that it does not specify any algorithm in particular. Its only requirement is the use of no-regret algorithms by all OD pairs (for an overview of some of them see [27]). In particular, we will consider the *Weighted Majority Algorithm* (WMA)[28], which originated in the context of online learning (more precisely from the *online prediction using expert advice* problem), and whose pseudo-code for OD pair k is described in Algorithm 1.

Algorithm 1 Weighted Majority Algorithm (WMA)

```
for t = 1, ..., ∞ do
    Obtain path costs φ_p ∀p ∈ P_k
    for every path p ∈ P_k do
        if φ_p > min_{q∈P_k} φ_q then
            r_p^k ← β × r_p^k
        end if
    end for
    Normalize the r_p^k
end for
```

At each iteration t, those paths whose cost is bigger than the minimum are punished by multiplying their respective r_p^k by a certain constant $\beta < 1$ (throughout our simulations we have used $\beta = 0.95$, a value that we empirically verified obtains very good results). Actually, and in order to avoid unnecessary changes in the traffic distribution, we shall only update r_p^k when the corresponding path cost is bigger than the minimum cost plus a certain margin (in the case of MinUG we fixed the margin at 0.005, and for MinDG we used 5% of the minimum).

5. A Preliminary Comparison

In this section we shall present some first simulations that will help us to gain insight into the mechanisms and highlight some of their respective shortcomings. Before, we will discuss how we performed these and the rest of the simulations.

As the reference network we used Abilene, a high-speed Internet2 backbone network. Abilene consists of 12 router-level nodes connected by 30 optical links (we only consider intra-domain links). The used router-level network topology and traffic demands are available at [29]. Traffic data consists of 6 months of traffic matrices collected every 5 minutes via Netflow from the Abilene Observatory [30]. As measured traffic demands do not significantly load the network, we re-scaled them by multiplying all their entries by a constant. The dataset in [29] also provides the static routing configuration R_o deployed in Abilene during the 6-month long TMs measurement campaign.

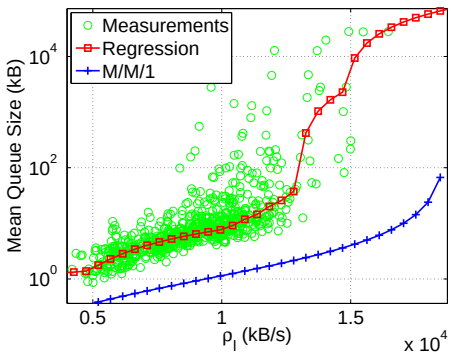


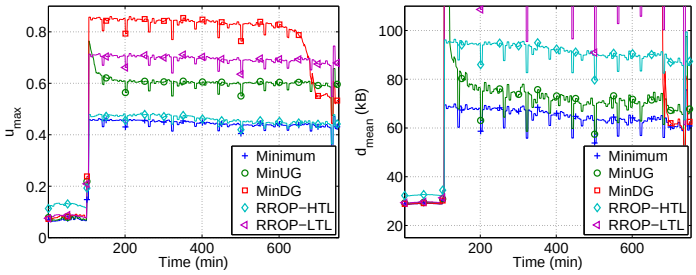
Figure 2: Mean queue size: measurements and approximations

In the case of MinDG, $f_l(\rho_l)$ is typically chosen based on a simplistic model (e.g. the M/M/1 model which yields $f_l(\rho_l) = \rho_l/(c_l - \rho_l)$) [9]. In order to avoid such arbitrary and unprecise choice, in [19] we proposed instead to *learn* this function (and its derivative) from measurements. Figure 2 depicts the real mean queue size of an operational network link at Tokyo obtained from [31], together with the *M/M/1* estimation $f_l^{M/M/1}(\rho_l)$ and the non-parametric regression $\hat{f}_l(\rho_l)$. It is clear that $f_l^{M/M/1}(\rho_l)$ consistently underestimates the real queue size value, while $\hat{f}_l(\rho_l)$ provides quite accurate results.

To be as fair as possible, all mechanisms use the same set of paths, namely those calculated by SRR as discussed in Sec. 3. The TMs are fed to the mechanisms in consecutive temporal order. Both DLB mechanisms (MinUG and MinDG) are initiated at arbitrary values of r_p^k, which will be updated as new link load measurements arrive. We have assumed that each OD pair receives these measurements every minute, meaning that for each new TM five updates of their corresponding r_p^k values will be performed (recall that TMs are collected every 5 min). Results are shown then for every minute. As a reference, we also computed the optimum values $u_{\max}^{\text{opt}}$ and $d_{\text{mean}}^{\text{opt}}$ for every TM X of the dataset.

In this example we consider a traffic scenario that presents an abrupt and large volume increase due to an external routing modification. This corresponds to the TMs with indexes between 1050 and 1200 from dataset X23 in [29]. The evaluation starts with a normal low traffic load situation, but after the 100th minute one of the OD flows abruptly increases its traffic volume, loading the links it traverses until the end of the evaluation.

Regarding SRR, in what follows we shall use the term RROP as a reference to SRR, recalling that the robust routing optimization problem is the one described in Table 1. Based on the static routing matrix of Abilene R_o we define two different polytopes, the former adapted to the



(a) Maximum link utilization (b) Mean e2e queuing delay

Figure 3: Maximum link utilization and mean end-to-end queuing delay. Traffic demand volume abruptly increases after the 100th minute.

Low Traffic Load period (LTL period, before the 100th minute) and the latter adapted to the High Traffic Load period (HTL period, after the 100th minute):

$$\mathbb{X}^{\text{LTL}} = \{X \in \mathbb{R}^m, R_o.X \leqslant Y^{\text{LTL}},\ X \geqslant 0\}$$
$$\mathbb{X}^{\text{HTL}} = \{X \in \mathbb{R}^m, R_o.X \leqslant Y^{\text{HTL}},\ X \geqslant 0\}$$

We assume that traffic is known in advance in both definitions, and take Y^{LTL} and Y^{HTL} as the maximum link load values observed during the LTL and HTL periods respectively. We compute two different robust routing configurations for both polytopes; RROP-LTL corresponds to the SRR configuration for polytope $\mathbb{X}^{\text{LTL}}$, and RROP-HTL for polytope $\mathbb{X}^{\text{HTL}}$. As we mentioned before, both RROP-LTL and RROP-HTL use the same set of paths, namely the paths obtained from Table 1 for polytope $\mathbb{X}^{\text{LTL}}$. Solving RROP for a given set of paths consists of only adding new extreme points of polytope $\mathbb{X}$ (i.e., only new constraints are added).

Results are presented in Fig. 3, which depicts (a) the maximum link utilization $u_{\max}$ and (b) the mean end-to-end queuing delay d_{mean} during the evaluation period. Let us first focus the attention on the performance of RROP-HTL after the 100th minute. Despite achieving an almost optimal performance as regards $u_{\max}$ (a relative difference with respect to the optimum smaller than 4%), RROP-HTL obtains a queuing delay that constantly exceeds the optimum by almost 40% under a moderate network load. Such a difference may not be even acceptable from a QoS perspective, where end-to-end delays are even more important than network congestion. As we will show later, this loss in performance is a direct consequence of the local criterion used in RROP.

A second interesting observation comes from the difference between RROP-HTL and RROP-LTL performances before and after the abrupt traffic volume increase; Fig. 3(a) shows that, despite an almost negligible network load, RROP-LTL outperforms RROP-HTL by almost 50% of relative utilization during the LTL period, while the opposite happens during the HTL period. The difference is not that big as regards delay before the 100th minute, but it becomes significant after the volume increase, where RROP-LTL obtains a very bad performance. These results are somehow expected given the polytopes definition, and brings to light both the dependence of RROP on the uncertainty set definition and the inherent consequence of using a single static routing configuration under large traffic variations. Let us highlight the fact that in this example we have considered that traffic was known in advance for the definition of both polytopes $\mathbb{X}^{\text{LTL}}$ and $\mathbb{X}^{\text{HTL}}$. While traffic during the LTL period is easy to predict, the definition of $\mathbb{X}^{\text{HTL}}$ in a real traffic scenario is a challenging task. We will come back to this issue in the following section.

Let us now discuss the results obtained by the dynamic schemes. A first important observation is that they present an important overshoot, with an absolute difference with the optimum $u_{\max}^{\text{opt}}$ of approximately 40%. Regarding MinDG in particular, convergence after the anomaly is very slow, taking more than 600 sec. However, it should be noted that when it eventually converges, it obtains a d_{mean} that is very similar to the optimum. In terms of $u_{\max}$, the difference with respect to the optimum is approximately 10%.

Special attention deserves the case of MinUG. After a shorter convergence time (approximately 100 sec.), the resulting value of $u_{\max}$ is not the optimum. Let us recall that this kind of game (which models schemes such as REPLEX of TeXCP) is used to converge to a routing configuration that minimizes the maximum link utilization [10, 11]. However, in this case, the difference is more than 15%. Both of these problems will be further discussed in the following section.

6. Improving the Algorithms Performance

The simple evaluation conducted in the previous section shows some conception drawbacks of the SRR and DLB algorithms presented in Sec. 3 and 4 respectively. In this section we shall explain the origin of these problems and present enhanced mechanisms to overcome them.

6.1. Improving Stable Robust Routing

6.1.1. Network-Wide Performance

As we showed in Fig. 3(b), the minimization of $u_{\max}$ leads to a distribution of traffic that results in an excessive end-to-end delay. Using the mean delay d_{mean} as the objective function in RROP (cf. Table 1) would be an interesting approach to ease the problem; however, $f_l(\rho_l)$ is a non-linear function and the optimization problem becomes too difficult to solve. As we previously said, optimization under uncertainty is more complex than classical optimization and simple optimization criteria should be used. In this sense, we could use instead the mean link utilization u_{mean} as the objective function.

The mean link utilization considers at the same time the load of every link in the network and not only the utilization of the most loaded link; as we will show in the results, such an objective function provides a better global

$$
\begin{array}{rcll}
\text{minimize} & u_{\text{mean}} & & \\
\text{subject to:} & & & \\
\sum_{l\in L}\sum_{k\in N}\sum_{p\in P_k}\frac{1}{c_l}\lambda_l^p\,.\,r_p^k\,.\,x(k) & \leqslant & u_{\text{mean}}.q & \forall\, X\in\mathbb{X} \\
\sum_{k\in N}\sum_{p\in P_k}\lambda_l^p\,.\,r_p^k\,.\,x(k) & \leqslant & u_{\text{max}}^{\text{thres}}.c_l & \forall\, l\in L,\ \forall\, X\in\mathbb{X} \\
\sum_{p\in P(k)} r_p^k & = & 1 & \forall\, k\in N \\
r_p^k & \geqslant & 0 & \forall\, p\in P_k,\ \forall\, k\in N
\end{array}
$$

Table 2: Robust Routing Mean Utilization Optimization Problem (RRMP)

$$
\begin{array}{rcll}
\text{minimize} & u_{\text{aof}}=\alpha\,.\,u_{\text{max}}+(1-\alpha)\,.\,u_{\text{mean}} & & \\
\text{subject to:} & & & \\
\sum_{l\in L}\sum_{k\in N}\sum_{p\in P_k}\frac{1}{c_l}\lambda_l^p\,.\,r_p^k\,.\,x(k) & \leqslant & u_{\text{mean}}.q & \forall\, X\in\mathbb{X} \\
\sum_{k\in N}\sum_{p\in P_k}\lambda_l^p\,.\,r_p^k\,.\,x(k) & \leqslant & u_{\text{max}}.c_l & \forall\, l\in L,\ \forall\, X\in\mathbb{X} \\
\sum_{p\in P(k)} r_p^k & = & 1 & \forall\, k\in N \\
r_p^k & \geqslant & 0 & \forall\, p\in P_k,\ \forall\, k\in N
\end{array}
$$

Table 3: Robust Routing AOF Optimization Problem (RRAP)

performance as regards end-to-end delay. However, a direct minimization of u_{mean} does not assure a bounded maximum link utilization, which is not practical from an operational point of view. In this sense, we propose to change the objective function in RROP by u_{mean}, while bounding the maximum link utilization by a certain threshold $u_{\text{max}}^{\text{thres}}$ defined a priori. The resulting problem, which we shall call the *Robust Routing Mean Utilization Optimization Problem* (RRMP), is defined in Table 2.

RRMP is solved in the same way as RROP, using the same recursive algorithm proposed in [6]. Note that the difference between the two problems is only a new constraint per each new traffic demand in $\mathbb{X}$ (in fact, for each extreme point of $\mathbb{X}$). The drawback of RRMP is its dependence on the value of $u_{\text{max}}^{\text{thres}}$, which directly influences the routing performance as we will shortly see. An interesting choice for $u_{\text{max}}^{\text{thres}}$ would be to use the output of RROP, namely $u_{\text{max}}^{\text{robust}}$. To some extent this would result in a similar routing solution but with better traffic balancing.

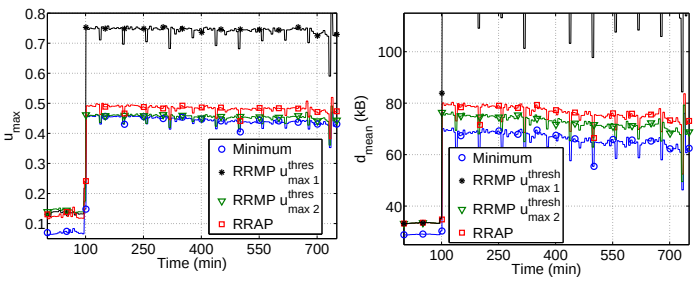
An alternative approach is to minimize both the value of u_{max} and u_{mean} at the same time, which constitutes a problem of multi-objective optimization (MOO). MOO problems are generally more difficult to solve because traditional single-objective optimization techniques cannot be directly applied. Nevertheless, the problem of finding all the Pareto-efficient solutions to a linear MOO problem is well known and different approaches can be used to treat the problem [32, 33]. In this work we consider an intuitive and easy approach to solve a MOO problem with standard single-objective optimization techniques. The approach consists in defining a single aggregated objective function (AOF) that combines both objective functions. We define a weighted linear combination of u_{max} and u_{mean} as the new objective function $u_{\text{aof}}=\alpha\,.\,u_{\text{max}}+(1-\alpha)\,.\,u_{\text{mean}}$, where $0\leqslant\alpha\leqslant 1$ is the combination fraction. Despite its simple form, this new objective is very effective and provides accurate results for both performance indicators. We shall call this new optimization problem as *Robust Routing AOF Optimization Problem* (RRAP), defined in Table 3. As before, RRAP is solved with the same algorithms used in RROP.

6.1.2. Comparison between RRMP and RRAP

We will now evaluate both the RRMP and RRAP versions of SRR in the traffic scenario previously considered in Sec. 5. In order to appreciate the dependence of RRMP on the maximum link utilization threshold $u_{\text{max}}^{\text{thres}}$, two different thresholds are used in the evaluation: $u_{\text{max}_1}^{\text{thres}}=1$ (which corresponds to the constraint $u_{\text{max}}\leqslant 1$ in Table 1), and $u_{\text{max}_2}^{\text{thres}}=u_{\text{max}}^{\text{robust}}$, where $u_{\text{max}}^{\text{robust}}$ is the output of RROP-HTL in Sec. 5. In the case of RRAP, the weight α is set to 0.5, namely an even balance between u_{max} and u_{mean}. This may impress as a somewhat naive approach to the reader, but practice shows that this choice provides in fact very good results.

Figures 4 depicts the results in this case. Let us focus our attention on the operation after the 100th minute, as all robust routing configurations use $\mathbb{X}^{\text{HTL}}$ as the uncertainty set. To be as fair as possible, both RRMP and RRAP use the same set of paths as those used by RROP in Fig. 3. The figure clearly shows that the performance of RRMP strongly depends on the threshold $u_{\text{max}}^{\text{thres}}$. In the case of $u_{\text{max}_1}^{\text{thres}}$, the attained maximum link utilization is well beyond the optimal values, reaching almost a 70% of relative performance degradation. This overload directly translates into huge mean end-to-end queuing delays. Results are quite impressive when considering the second threshold, both as regards u_{max} and d_{mean}. RRMP using $u_{\text{max}_2}^{\text{thres}}$ provides a highly efficient robust routing configuration, showing that it is possible to improve current implementations of SRR with a slight modification of the objective function. However, this dependence on the threshold $u_{\text{max}}^{\text{thres}}$ introduces a new tunable parameter, something undesirable when looking for solutions that simplify network management.

As regards RRAP, obtained results are slightly worse than those obtained by RRMP $u_{\text{max}_2}^{\text{thres}}$, but still very close to the optimal performance, with a relative performance degradation of about 10% as regards u_{max} and d_{mean} with respect to an optimal routing configuration. Nevertheless, RRAP has no tunable parameter apart from the combination factor α, which in fact is set to a half independently of the traffic situation.



(a) Maximum link utilization (b) Mean e2e queuing delay

Figure 4: Maximum link utilization and mean end-to-end queuing delay for RRMP and RRAP.

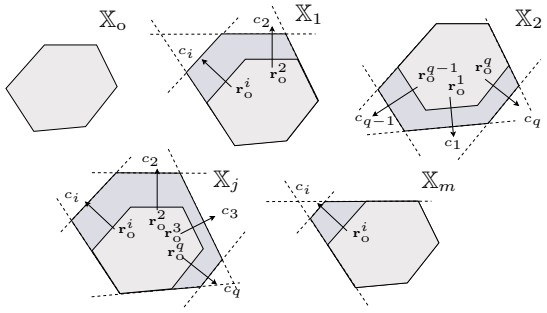


Figure 5: Different anomaly polytopes for preemptive robust routing computation.

6.1.3. The Reactive Robust Routing

As we showed in Sec. 5, the definition of the uncertainty set has a major impact on the performance of SRR. In particular, we saw that using a single definition of uncertainty set under highly variable traffic cannot provide routing efficiency for both normal operation traffic and unexpected traffic events. Despite being one of its most important features, using a single SRR configuration is not the best strategy.

In [13] we proposed an adaptive version of SRR, known as the *Reactive Robust Routing* (RRR). The basic idea in RRR consists in computing a primal robust routing configuration R^{o}_{robust} for expected traffic variations in normal operation within a primal polytope $\mathbb{X}_o$. This polytope is defined as in Sec. 3, based on a certain fixed routing configuration R_o and the expected links traffic load we shall call $Y_o = \{\rho_{o_i}\}$. Additionally, a set of m *anomaly* polytopes $\mathbb{X}_j$ are defined, and a preemptive robust routing configuration R^{j}_{robust} is computed for each of these anomaly polytopes.

Let us explain the concept of an anomaly polytope. In Fig. 3, the abrupt increase in traffic volume is caused by a single *anomalous* OD flow x_k that unexpectedly carries a many times bigger traffic load θ due to an external routing modification. After this exogenous unexpected event, the traffic demand X takes the value $X' = X + \theta.\boldsymbol{\delta}_k$, where $\boldsymbol{\delta}_k = (\delta_{1,k}, \ldots, \delta_{k,k}, \ldots, \delta_{m,k})^T$, $\delta_{i,k} = 0$ if $i \neq k$ and $\delta_{k,k} = 1$. We shall designate this unexpected traffic increase in OD flow x_k as anomalous traffic event A_k. The anomaly polytope $\mathbb{X}_k$ results from expanding the primal polytope $\mathbb{X}_o$ in the directions of the links that traverses the anomalous OD flow x_k, with respect to R_o. The reader should bear in mind that the kind of unexpected traffic events we deal with are independent of the intradomain routing; these events originate outside the network and propagate between origin-destination nodes. This justifies the relevance of the polytope expansion with respect to R_o. The obtained polytope $\mathbb{X}_k$ is the smallest polytope that contains the unexpected traffic demand X' and thus, the corresponding robust routing configuration R^{k}_{robust} provides a relatively good performance under its occurrence. Figure 5 explains the idea of the multiple anomaly polytope expansion. As before, $\mathbf{r}^{i}_{o}$ stands for the i-th row of the routing matrix R_o.

Note that in a real scenario it is not possible to predict the size of the anomalous traffic θ. As a consequence, the primal polytope $\mathbb{X}_o$ is expanded to the limits of link capacities, obtaining the following anomaly polytope for each anomalous traffic event A_k:

$$\mathbb{X}_k = \left\{ X \in \mathbb{R}^m, R_o.X \leqslant Y^{A_k},\ X \geqslant 0 \right\},\ \forall k \in N \qquad (6)$$

In (6), the i-th component of Y^{A_k} takes the value ρ_{o_i} if $r^{i,k}_{o} = 0$, or the value c_i if $r^{i,k}_{o} > 0$, being $r^{i,k}_{o}$ the element (i,k) of R_o.

Given the primal and the m preemptive robust routing configurations R^{o}_{robust} and R^{j}_{robust}, RRR uses an on-line anomaly detection/localization sequential algorithm to detect the occurrence of an anomalous event A_k, switching routing from R^{o}_{robust} to R^{k}_{robust} (and from R^{k}_{robust} back to R^{o}_{robust} when normal operation is regained). We refer the reader to [13] for additional details on the detection/localization algorithms and the implementation of RRR.

RRR can handle large and unexpected traffic variations in single OD flows quite effectively (the case of multiple simultaneous anomalies is beyond the scope of RRR). However, given the difficulty involved in modifying the routing configuration of a large scale network in an on-line fashion, the contributions of RRR are mainly theoretical. This problem can be solved by using a load balancing technique instead of a complete routing reconfiguration. In load balancing, we keep the same set of paths P_k for each OD pair k, and only modify the fractions of traffic sent along each path. Load balancing can be easily performed on-line and does not require any additional modifications in current path-based networks such as MPLS. We shall refer to the load balancing variant of RRR as *Reactive Robust Load Balancing* (RRLB), stressing the difference between routing reconfiguration and load balancing.

RRLB uses the same set of anomaly polytopes $\mathbb{X}_j$ defined in RRR, but the computation of the m preemptive robust routing configurations R^{j}_{robust} is slightly modified. The same set of paths P_k obtained during the computation of R^{o}_{robust} is used in every R^{j}_{robust}. As in Sec. 5 and 6.1.2,

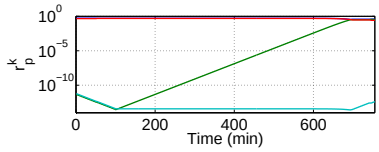


Figure 6: Evolution of r_p^k for the anomalous OD pair (MinDG)

routing configurations R^j_{robust} are obtained with a simplified version of the former optimization algorithm, where only new traffic demands are progressively added and no extra paths are created.

6.2. Improving Dynamic Load Balancing

6.2.1. Convergence Time

The DLB algorithms evaluated in Sec. 5 present an important overshoot and a significant settling time in the presence of sudden and large traffic variations. If the traffic anomaly is a perfect step, then the overshoot is unavoidable. We will try to address the long settling time instead. The reason behind this problem is relatively simple, as shown in Fig. 6. The graph depicts the evolution over time of the corresponding r_p^k values of the anomalous OD pair for MinDG in the example of Sec. 5. We may see that, although the r_p^k change exponentially fast, at the moment of the anomaly the values that should increase are so small that it takes them a very long time to converge. A possible solution is to impose a minimum value to all r_p^k. However, this will affect the precision of the algorithm and will still result in significant settling times.

Actually, r_p^k may be regarded as an indicator of the performance of path p in the previous iterations. A very small r_p^k means that p performed very badly with respect to the rest of the paths in the past. However, when the anomaly occurs, conditions severely change and history is no longer as relevant. If we consider that we are in such situation, we could for instance completely ignore history and restart the game by setting $r_p^k = 1/|P_k| \, \forall k \in P_k$. Before deciding how to reassign r_p^k, we will discuss how an OD pair may decide if it should restart its game or not.

Consider a situation where most of the traffic for OD pair k is routed along a path that is not the cheapest, and that the r_p^k corresponding to the minimum-cost path is very small. This could mean that although the former performed better in the past, this is no longer true and some traffic should be re-routed to the latter. This is more so as the difference in cost increases. However, this "suspicious" situation could be due to noisy measurements. To make sure that the game has actually changed and that it should be restarted, we will require such a situation to persists during a certain number of consecutive iterations. Once we detected that the game should be restarted, we will re-route some of the traffic that was being routed along the path with the biggest r_p^k to the cheapest one. The amount will be proportional to the relative difference in cost to avoid overreacting. Finally, remember that with WMA fast adaptation is achieved when the r_p^k are not too small. The objective with this "game restart" is simply to move r_p^k from critically small values. The algorithm will then rapidly converge to the optimum. We now present the pseudo-code of the complete algorithm for OD pair k:

Algorithm 2 WMA with Restart (WMA-R)

for $t = 1, \ldots, \infty$ **do**
  Obtain path costs $\phi_p \; \forall p \in P_k$
  Determine $p_{\min} = \underset{p \in P_k}{\operatorname{argmin}} \, \phi_p$ and $p_{\max} = \underset{p \in P_k}{\operatorname{argmax}} \, r_p^k$
  if $(r^k_{p_{\min}} < 0.1)$ and $(\phi_{p_{\min}} + \phi_{th} < \phi_{p_{\max}})$ **then**
    $n_e^k \leftarrow n_e^k + 1$
  else
    $n_e^k \leftarrow 0$
  end if
  if $n_e^k \leq n_{th}^k$ **then**
    Perform a normal iteration of WMA (cf. Algorithm 1)
  else
    $n_e^k \leftarrow 0$
    $\Delta_r \leftarrow \min\left\{\frac{\phi_{p_{\max}}}{\phi_{p_{\min}}} - 1, 1\right\} \times \frac{r^k_{p_{\max}} - r^k_{p_{\min}}}{2}$
    $r^k_{p_{\max}} \leftarrow r^k_{p_{\max}} - \Delta_r$
    $r^k_{p_{\min}} \leftarrow r^k_{p_{\min}} + \Delta_r$
  end if
end for

The new variable n_e^k counts the number of consecutive occurrences of a "suspicious" situation (we used $n_{th}^k = 3$). The threshold ϕ_{th} is to make sure that the difference in cost between paths is significant. In particular, for MinUG we used $\phi_{th} = 0.005$ and for MinDG $\phi_{th} = 0.2\phi_{p_{\min}}$. Finally, note that when the game is restarted, we re-route a certain amount of traffic from $p_{\max}$ to $p_{\min}$, but at most the amount of traffic routed along each path is equalized.

6.2.2. Converging to the Social Optimum in Bottleneck Games

In Sec. 5 we showed an example in which MinUG does not converge to the optimum, and obtains a difference of 15% with respect to the optimum MLU. The reason behind this poor performance is simply that MinUG does not take into account the result regarding the optimality of the WE and the *efficiency condition* discussed in Sec. 4.1 and originally presented in [15]. This result states that if at a WE all OD pairs send their traffic along paths with a minimum number of network bottleneck links (those with the maximum utilization in the whole network), the WE is optimal. The problem we analyze now is how to design a path cost function ϕ_p that takes into account this condition, so that when using it, the load-balancing algorithm converges, when possible, to the correct WE. Note that the condition is only sufficient, meaning that a WE that fulfills the efficiency condition may not exist. A simple example of such case is a single OD pair with two paths with different lengths, where all links have the same capacity. Anyhow, the two main difficulties in the design of such path cost are the following. Firstly, the number of bottleneck links in a path is an integer (thus not continuous on

r_p^k). Secondly, the probability of two links having exactly the same utilization is zero, and as such we should consider the number of links that have an utilization *similar* to the network bottleneck.

The objective is then to find a cost function that penalizes paths in which several links have similar utilizations (and that this utilization is the maximum in all the network), and that it does not switch between values to avoid oscillations. A candidate ϕ_p that fulfills these two conditions is the so-called *log-sum-exp* function. Consider a set of arbitrary numbers $A = \{a_i\}$, the log-sum-exp function $g(A)$ is defined as follows:

$$g(A) = \frac{1}{\gamma_A} \log \left(\sum_{i=1,\ldots,|A|} e^{\gamma_A a_i} \right) =$$

$$a_{i^*} + \frac{1}{\gamma_A} \log \left(1 + \sum_{i=1,..,|A| \wedge i \neq i^*} e^{\gamma_A (a_i - a_{i^*})} \right) \quad (7)$$

Consider the special case in which $a_{i^*} = \max A$. It should be clear that if a_{i^*} is significantly bigger than the rest of the elements in A, the above convex and non-decreasing function constitutes an excellent approximation of a_{i^*}. In fact, it easy to prove that $a_{i^*} \leq g(A) \leq a_{i^*} + \log(|A|)/\gamma_A$, meaning that we may control the precision of the approximation through the parameter γ_A (the bigger this parameter, the more precise the resulting approximation). Moreover, as more elements in A are similar to the maximum, $g(A)$ approaches the upper bound, reaching it when all elements are the same.

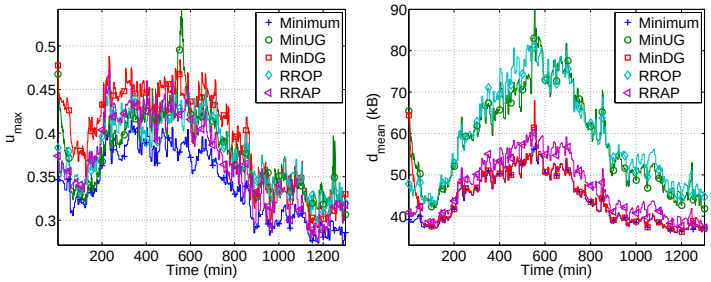
We will then use the second term of (7) as a penalty to those paths with several links whose utilization is similar to $u_{\max}$ (the maximum utilization in the network). More precisely, given a path p, let $U_p = \{u_l\}_{l \in p}$ be the utilizations in the path, and $l^* \in p$ be the link with the biggest utilization in p. We will then use the penalty function with the alternative set U_p^*, which has the same elements as U_p, but substitutes u_{l^*} by $u_{\max}$. This results in the following cost function:

$$\phi_p = u_{l^*} + \frac{1}{\gamma_p} \log \left(1 + \sum_{l \in p \wedge l \neq l^*} e^{\gamma_p (u_l - u_{\max})} \right) \quad (8)$$

Even if this new cost function penalizes paths with several network bottleneck links, it also penalizes longer paths, which was not our original objective. A good choice of γ_p will alleviate this side-effect. For instance, we used $\gamma_p = \log(|p|)/\max\{0.01, u_{l^*}/10\}$. This way, we try to minimize the effect of $\log(|p|)$ and relativize the penalization to u_{l^*}. The following section presents the results obtained by this new path cost function, which we will still call MinUG.

7. Evaluation and Discussion

In this section we evaluate the performance of the different RR and DLB algorithms presented in this work, considering both normal operation and anomalous traffic situations. We present and discuss three simulation case-scenarios: starting from a normal traffic variation scenario, we increase the number of OD pairs that present anomalous traffic variations. This allows for performance comparison at different levels of traffic variability. As both RRAP and RRMP provide similar results (when $u_{\max}^{\text{thres}}$ is correctly defined for RRMP), we will only consider the RRAP mechanism in the evaluation. Finally, we shall use RRLB-OP and RRLB-AP to designate the Reactive Routing Load Balancing variants of RROP and RRAP respectively.



(a) Maximum link utilization (b) Mean e2e queuing delay

Figure 7: Maximum link utilization and mean end-to-end queuing delay under normal operation.

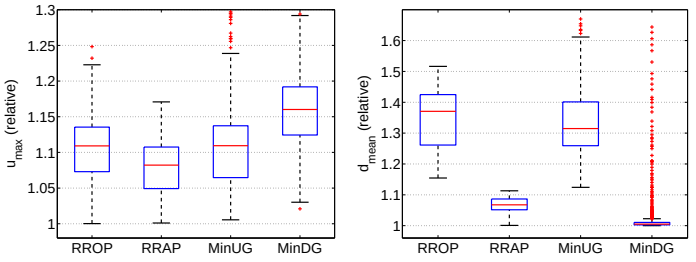
7.1. Normal Operation

The first case-scenario corresponds to traffic in normal operation. The only variability is due to typical daily fluctuations. Figure 7 presents the evolution of $u_{\max}$ and d_{mean} for RROP and RRAP, using a set of 260 TMs from dataset X01 in [29] (specifically, those with indexes between 420 and 680). All algorithms perform similarly as regards maximum link utilization, depicted in Fig. 7(a). This may be further appreciated in the boxplot summary presented in Fig. 8(a), where values are relative to those obtained with an optimal routing configuration. Note that the relative performance degradation is around 10% in most cases.

Figures 7(b) and 8(b) show that results are quite different as regards mean queuing delay. We may verify that the best results are obtained by MinDG, followed closely by RRAP. However, both RROP and MinUG systematically obtain a significant difference with respect to the optimum, generally between 30% and 40%. These results further highlight the limitations of RROP and MinUG as previously discussed: using $u_{\max}$ as a performance objective results in a relatively low maximum utilization, but neglects the rest of the links, impacting the network-wide performance.

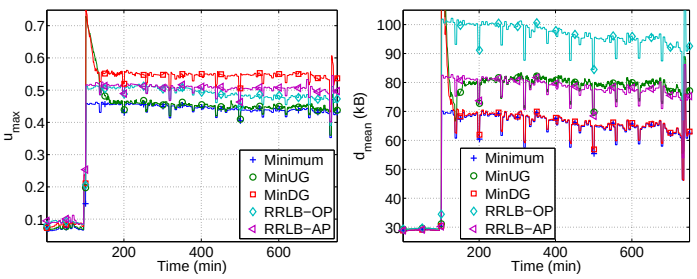
7.2. One Anomalous OD Pair

The second case-scenario is the one considered in Sec. 5, where there is a sudden and abrupt increase of the traffic



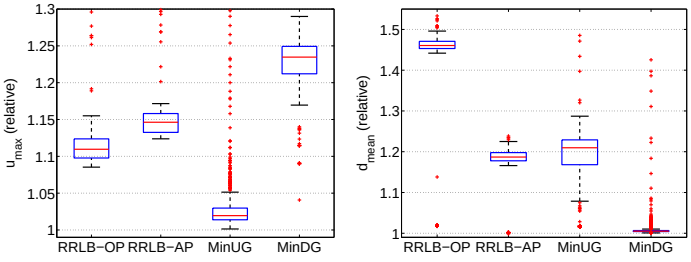
(a) Relative maximum link utilization (b) Relative e2e queuing delay

Figure 8: Maximum link utilization and mean end-to-end queuing delay under normal operation, boxplot performance summary. Depicted results are relative to the optimal values.



(a) Maximum link utilization (b) Mean e2e queuing delay

Figure 9: Maximum link utilization and mean end-to-end queuing delay under one anomalous OD pair.



(a) Relative maximum link utilization (b) Relative e2e queuing delay

Figure 10: Maximum link utilization and mean end-to-end queuing delay under one anomalous OD pair, boxplot performance summary. Depicted results are relative to the optimal values.

volume carried by one OD flow. As a difference with respect to the evaluation in Fig. 3, where traffic was assumed known in advance, this case-scenario corresponds to a real situation where traffic anomalies can not be forecast.

Firstly, notice in Fig. 9 how the improvements discussed in Sec. 6.2.1 for MinUG and MinDG result in a relatively smaller overshoot than before, but most importantly the settling time has been significantly decreased (in the case of MinDG, from 600 min. to less than 50 min). Moreover, note how the modified cost function proposed in Sec. 6.2.2 results in MinUG converging to the socially optimum WE.

Regarding $u_{\max}$, both RRLB-OP and RRLB-AP obtain similar results, with a relative performance degradation generally smaller than 15%. Note that while relatively important, this performance degradation is surprisingly small if we consider that traffic increases more than 500% in less than 10 minutes. The same may be said about MinDG, which obtains a degradation between 20% and 25%. In terms of d_{mean}, MinUG and RRLB-AP perform similarly. They both clearly outperform RRLB-OP, achieving a relative mean queuing delay almost 30% smaller. These results reinforces once again our observations about the difficulty in RROP to attain global performance, and the advantages of using a simple network-wide objective function in a robust routing algorithm. Moreover, they also illustrate the difference between MinUG and RROP. Even when MinUG was designed with the same objective than RROP (namely to minimize $u_{\max}$), the fact that in MinUG each OD pair greedily minimizes the path utilization results in a different overall behavior.

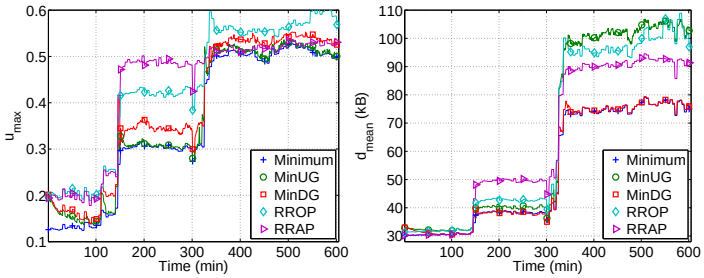
7.3. Two Anomalous OD Pairs

In this case-scenario two OD pairs largely increase their traffic demand, one at approximately the 150th minute and the other at the 320th (they correspond to TMs with indexes between 160 and 280 from dataset X06 in [29]). They both present this anomalous traffic until the end of the simulation. We shall then separate the simulation in three parts: the first third where traffic is normal, the second third were only one OD pair is anomalous, and the last third were both OD pairs are anomalous. The anomaly localization algorithm of RRR was designed for the case of one single anomalous OD pair. Because of this, we will further illustrate the tradeoff between size of the considered uncertainty set and efficiency of the obtained routing, and chose the uncertainty polytope by the traffic loads seen after the second anomaly.

In Fig. 11(a) we may see that, as expected, the $u_{\max}$ obtained by both RROP and RRAP in the last third of the simulation are very close to the optimum. However, in the rest of the simulation the difference may be important, specially in the second part where the absolute difference for RRAP is almost 20%. It is important to highlight the results obtained by MinDG and MinUG. Notice that the overshoot this time is much smaller than before (a maximum of 0.1 in $u_{\max}$ for MinDG) and the settling time is negligible. In this case, the increase in traffic of the anomalous OD pairs is more gradual than before, which clearly favors dynamic schemes in their performance.

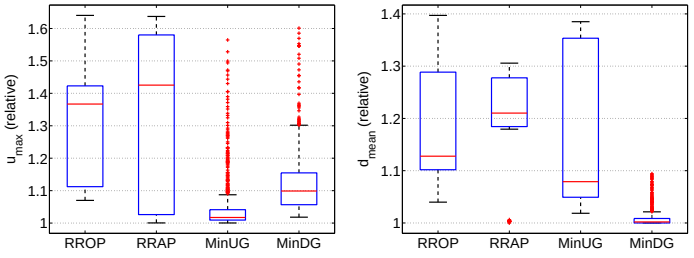
8. Conclusions and Future Work

From the study we presented in this article we may reach several conclusions. The most important is probably that we have shown that using a single routing configuration is not a cost-effective solution when traffic is relatively dynamic. Stable Robust Routing obtains a rather poor performance either when faced with non considered traffic demands (tight uncertainty sets) or when designed



(a) Maximum link utilization (b) Mean e2e queuing delay

Figure 11: Maximum link utilization and mean end-to-end queuing delay under two anomalous OD pair.



(a) Relative maximum link utilization (b) Relative e2e queuing delay

Figure 12: Maximum link utilization and mean end-to-end queuing delay under two anomalous OD pair, boxplot performance summary. Depicted results are relative to the optimal values.

to manage as many traffic demands as possible (big uncertainty sets). It is clear from our study that some form of dynamism is necessary, which could be either RRLB (Reactive Robust Load Balancing) or DLB (Dynamic Load-Balancing).

RRLB computes a nominal operation routing configuration, and has an alternative routing (using the same paths than in normal operation) for certain possible anomalous situations. In order to detect these anomalous situations, link load measurements have to be gathered [13]. On the other hand, DLB gathers these same measurements but also requires updating load-balancing in a relatively small time-scale. The added complexity is then to distribute these measurements to all ingress routers (instead of a central entity) and updating the load-balancing in real-time.

Our results show that the additional complexity involved in DLB is not justified when the variability (or the anomalies) are not very significant. However, the use of DLB under highly dynamic traffic is very appealing and generally provides better results than RRLB. Moreover, if the anomalies may not be correctly detected or localized (as in Sec. 7.3), the only effective solution is DLB.

Regarding RR in particular, we saw that using a local performance criterion such as the maximum link utilization (MLU) is not a suitable objective function as regards network-wide performance and QoS provisioning. In particular, we showed that an almost optimal robust routing configuration with respect to MLU can experience rather high mean end-to-end queuing delays, a very important performance indicator for all types of traffic. The maximum link utilization is widely used in current network optimization problems, particularly in most Robust Routing proposals, thus we believe that this simple evidence can help and should be considered in enhanced future implementations.

In fact, we have shown that objective optimization functions can be kept simple, and yet better network-wide performance can be attained. By using a simple combination of performance indicators such as the maximum and the mean link utilization, we obtained a robust routing configuration that definitely outperforms current implementations from a global end-to-end perspective, while achieving very similar results as regards worst-case link utilization.

The framework of Aggregated Objective Functions (AOF) we used provides interesting results as regards multi-objective optimization, particularly in the context of robust optimization. An AOF approach can be used to construct better objective functions from simple performance indicators, avoiding the need of more complex Multi-Objective Optimization (MOO) techniques. As part of our ongoing work we are currently analyzing the trade-off between using a simple AOF approach against a more complex but more complete MOO approach, computing all Pareto-efficient solutions for a polyhedral uncertainty set and comparing their performance.

In what respects DLB, dynamic approaches are generally met with reluctance due to their transient behavior under strong traffic variations. However, we have shown that this transient behavior can be effectively controlled, or at least alleviated, by simple mechanisms. Concerning the two different games we presented, conclusions are similar to those of RR. Striving to minimize d_{mean} instead of $u_{\max}$ results in a somewhat bigger maximum utilization, but a (sometimes much) better global performance.

It should also be highlighted that this article represents one of the first studies using no-regret algorithms for load-balancing, and as such much exploration is left to be done. In this sense, let us remark that WMA is arguably the simplest no-regret algorithm in the literature. For instance, it is easy to see that not all $\beta < 1$ guarantee a non-oscillatory behavior of the algorithm (consider for example $\beta = 0$). There exist other more sophisticated algorithms of this kind, that do not require any parametrization at all (including β) and still guarantee convergence (see for instance [34]), whose exploration is left for future work. However, an important conclusion of the current article is that, whatever the no-regret algorithm we choose, we will still require a "restart" feature as in WMA-R.

Our study also highlighted a problem with previously proposed DLB algorithms, namely the wrong assumption that OD pairs that greedily minimize the path utilization converge to a routing configuration that minimizes MLU. Based on a recent result [15], we have explored the pos-

sibility of modifying the path cost function so that the resulting routing configuration is actually optimum. Preliminary results presented in this article are very promising and encourage us to further study this new cost function (e.g. characterize the resulting WE or consider other alternative penalization functions).

9. Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments. This work was partially funded by CELTIC project TRANS and FP7 project Euro-NF.

[1] Cisco Systems, Global IP Traffic Forecast and Methodology 2006-2011, White Paper (2007 - updated 2008).

[2] Cisco Systems, The Exabyte Era, White Paper (2007 - updated 2008).

[3] Global Action Plan, An Inefficient Truth.
URL http://www.globalactionplan.org.uk/upload/resource-/Full-report.pdf

[4] M. Gupta, S. Singh, Greening of the Internet, in: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '03), Karlsruhe, Germany, 2003, pp. 19–26.

[5] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True, Deriving Traffic Demands for Operational IP Networks, in: IEEE/ACM Trans. on Networking, Vol. 9 (3), 2001, pp. 265–279.

[6] W. Ben-Ameur, H. Kerivin, Routing of Uncertain Traffic Demands, Opt. and Eng. 6 (3) (2005) 283–313.

[7] D. Applegate, E. Cohen, Making intra-domain routing robust to changing and uncertain traffic demands: understanding fundamental tradeoffs, in: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '03), Karlsruhe, Germany, 2003, pp. 313–324.

[8] H. Wang, H. Xie, L. Qiu, Y. R. Yang, Y. Zhang, A. Greenberg, COPE: traffic engineering in dynamic networks, SIGCOMM Comput. Commun. Rev. 36 (4) (2006) 99–110.

[9] A. Elwalid, C. Jin, S. Low, I. Widjaja, MATE: MPLS adaptive traffic engineering, in: IEEE Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. (INFOCOM 2001), Vol. 3, Anchorage, USA, 2001, pp. 1300–1309.

[10] S. Kandula, D. Katabi, B. Davie, A. Charny, Walking the tightrope: responsive yet stable traffic engineering, in: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '05), Philadelphia, USA, 2005, pp. 253–264.

[11] S. Fischer, N. Kammenhuber, A. Feldmann, REPLEX: dynamic traffic engineering based on wardrop routing policies, in: Proceedings of the 2006 ACM CoNEXT conference (CoNEXT '06), Lisboa, Portugal, 2006, pp. 1–12.

[12] A. Khanna, J. Zinky, The revised ARPANET routing metric, SIGCOMM Comput. Commun. Rev. 19 (4) (1989) 45–56.

[13] P. Casas, L. Fillatre, S. Vaton, Robust and Reactive Traffic Engineering for Dynamic Traffic Demands, in: Next Generation Internet Networks (NGI 2008), Krakow, Poland, 2008, pp. 69–76.

[14] A. Blum, E. Even-Dar, K. Ligett, Routing without regret: on convergence to nash equilibria of regret-minimizing algorithms in routing games, in: Twenty-Fifth Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC 2006), Denver, USA, 2006, pp. 45–52.

[15] R. Banner, A. Orda, Bottleneck Routing Games in Communication Networks, IEEE Journal on Selected Areas in Communications 25 (6) (2007) 1173–1179.

[16] M. Roughan, M. Thorup, Y. Zhang, Traffic engineering with estimated traffic matrices, in: Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement (IMC '03), Miami Beach, USA, 2003, pp. 248–258.

[17] C. Zhang, Y. Liu, W. Gong, J. Kurose, R. Moll, D. Towsley, On optimal routing with multiple traffic matrices, in: 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005), Vol. 1, Miami Beach, USA, 2005, pp. 607–618.

[18] P. Casas, S. Vaton, An Adaptive Multi Temporal Approach for Robust Routing, in: Euro-FGI Workshop on IP QoS and Traffic Control, Lisbon, Portugal, 2007.

[19] F. Larroca, J.-L. Rougier, Minimum-Delay Load-Balancing Through Non-Parametric Regression, in: Proceedings of the 8th International IFIP-TC 6 Networking Conference (NETWORKING '09), Aachen, Germany, 2009, pp. 782–794.

[20] M. Johansson, A. Gunnar, Data-driven traffic engineering: techniques, experiences and challenges, in: 3rd International Conference on Broadband Communications, Networks and Systems (BROADNETS 2006), San José, USA, 2006, pp. 1–10.

[21] I. Juva, Robust Load Balancing, in: IEEE Global Telecommunications Conference (GLOBECOM '07), Washington D.C., USA, 2007, pp. 2708–2713.

[22] R. K. Ahuja, T. L. Magnanti, J. B. Orlin, Network Flows: Theory, Algorithms, and Applications, Prentice Hall, 1993.

[23] D. Mitra, K. Ramakrishnan, A case study of multiservice, multipriority traffic engineering design for data networks, in: 1999 Global Telecommunications Conference (GLOBECOM '99), Vol. 1B, 1999, pp. 1077–1083.

[24] E. Altman, T. Boulogne, R. El-Azouzi, T. Jiménez, L. Wynter, A survey on networking games in telecommunications, Comput. Oper. Res. 33 (2) (2006) 286–311.

[25] F. Larroca, J.-L. Rougier, Routing Games for Traffic Engineering, in: IEEE International Conference on Communications (ICC '09), Dresden, Germany, 2009.

[26] J. Wardrop, Some theoretical aspects of road traffic research, Proceedings of the Institution of Civil Engineers, Part II 1 (36) (1952) 352–362.

[27] R. Yaroshinsky, R. El-Yaniv, S. S. Seiden, How to Better Use Expert Advice, Mach. Learn. 55 (3) (2004) 271–309.

[28] N. Littlestone, M. K. Warmuth, The weighted majority algorithm, Inf. Comput. 108 (2) (1994) 212–261.

[29] Yin Zhang, Abilene Dataset.
URL http://www.cs.utexas.edu/ yzhang/research/AbileneTM/

[30] The Abilene Observatory.
URL http://abilene.internet2.edu/observatory/

[31] K. Cho, WIDE-TRANSIT 150 Megabit Ethernet Trace 2008-03-18.
URL http://mawi.wide.ad.jp/mawi/samplepoint-F/20080318/

[32] J. Evans, R. Steuer, A Revised Simplex Method For Linear Multiple Objective Programs, Mathematical Programming 5 (1) (1973) 54–72.

[33] J. Ecker, I. Kouada, Finding All Efficient Extreme Points for Multiple Objective Linear Programming Programs, Mathematical Programming 14 (1978) 249–261.

[34] P. Auer, N. Cesa-Bianchi, C. Gentile, Adaptive and Self-Confident On-Line Learning Algorithms, Journal of Computer and System Sciences, vol. 64, 2002, 48–75.