Maximum Delay Computation for Interdomain Path Selection

Isabel Amigo, Sandrine Vaton, Thierry Chonavel and Federico Larroca

Abstract

One important problem when deploying interdomain path selection is advertising metrics that hold for a long period of time. In this paper we propose a method to aid interdomain path selection mechanisms in that sense. We present a means of computing a bound on the end-to-end delay of traversing a domain considering that the traffic varies within a given uncertainty set. This provides a robust and a verifiable quality of service value for traversing the Autonomous System (AS), without revealing confidential information. Consequently, the bound can be safely conceived as a metric to be announced by each AS in the process of interdomain path selection. We show how the maximum delay value is obtained for an interdomain bandwidth demand and we propose an exact method and a numerical approximation method for computing it, neither of which rely on a complex monitoring infrastructure. Simulations with real data that illustrate the problem and validate our results are also presented.

Keywords: 1.3. IP networks, 4.6. SLA management, 7.2. Optimization theories

1 Introduction

There is increasing interest in value-added services, such as videoconferencing or other bandwidth-on-demand services. In this context QoS and how to guarantee it becomes a crucial issue for all involved actors, i.e. the Network Provider, the Service Provider and the Customer. This is especially difficult

when the service traverses several domains, or Autonomous Systems (AS). In this case, QoS must be provided by all the ASs involved, which raises several technical, economical and political issues. Concerning the technical aspects, achieving scalability, preserving confidentiality and providing interoperability is paramount in any solution [1].

We will focus on point-to-point services with QoS requirements. In this case the service may be abstracted to a QoS guaranteed tunnel (for instance an MPLS tunnel [2]). The path must cross those domains through which destination is reached and whose combination of QoS parameters fulfills the service requirements.

In the framework of an alliance of ASs, carriers are envisioned to work together in order to achieve a common interest. In this scenario QoS values related to each domain are exchanged, and Traffic Engineering decisions are taken after them. Different mechanisms have been proposed for the selection and establishment of interdomain QoS-constrained tunnels, that mainly rely on RSVP-TE [3] and the PCE architecture [4] (e.g. [5, 6, 7, 8]). These mechanisms are based on metrics announced by each AS but they do not specify how to compute such metrics. In any case, the announced metrics have to hold for some period of time, ideally as long as the service is provided. Hence, it is interesting for ASs to be able to provide QoS values that are guaranteed to hold for a certain period of time.

Other approaches providing methods for end-to-end QoS can be found in the literature. For instance, some propose extensions to the de facto standard interdomain routing protocol BGP (see for instance [9]). Others propose ad-hoc functions to BGP, like [10, 11]. These are based on self-adaptive methods and perform routing decisions at the edge routers level in order to maintain certain QoS parameters below some given bounds. They monitor the network state obtaining feedback which acts as an input to the self-adaptive engine. These methods are conceived to work in a pure BGP network. However, we are interested in the case of explicitly signaled tunnels, like the PCE-based mechanisms, since they are more suitable in the context we are working on. For instance, we seek a method that strictly achieves the QoS needed and not only soft QoS. In addition, for reasons explained below, we seek for a method with light dependence on monitoring.

For certain services, available bandwidth and end-to-end delay are critical parameters. The latter is composed of the sum of the delays introduced by each transit AS and the terminal ones, from source to destination. As illustrated in Fig. 1, where we show a situation with two terminal ASs and

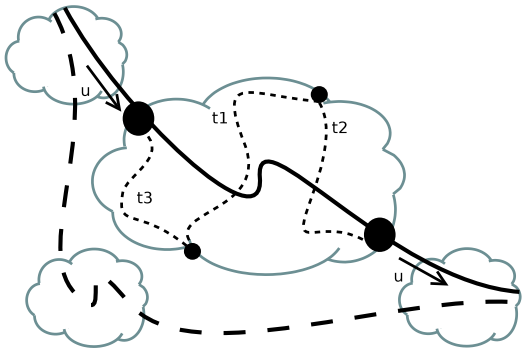


Figure 1: Scenario

one transit AS, the delay in each of the ASs depends on the traffic already present in the AS (t_* flows in Fig. 1), the topology, the routing configuration, and the traffic coming from the new tunnel (flow u in Fig. 1).

Naively, we may think that the problem of choosing the delay value to advertise can be reduced to simply advertising the current one. However, this presents two main problems, as commented in the following.

Monitoring the delay is itself a complex task. Several techniques have been proposed in the literature, mainly based on passive measurements, where some packets are timestamped and sent to a collector (see for instance [12, 13]), or on active ones, where probe packets are sent along the network and the delay is inferred from the one experienced by the probe packets (see for instance [14, 15]). These techniques present several drawbacks, just to mention the most common of them, they usually present issues of bandwidth consumption and need for synchronization, for instance, based on specific equipment as GPS devices. Moreover, all techniques need for a monitoring architecture, which can become complex when accuracy is needed [16].

In addition, even if we were able to accurately measure the delay, the announced value, as mentioned above, should hold for a certain period of time. In this scenario the complexity is mainly due to the existence of uncertainty. This uncertainty can be classified into two types: network state uncertainty and traffic uncertainty.

Uncertainty in network state refers to the situation where the topology changes or is partially known. This may be due to information arriving out of date or not synchronized to the entity performing the computation, or simply

to link failures. In the literature some approaches have been proposed for performing QoS routing under this kind of uncertainty [17, 18, 19]. However, in the present paper we will assume that the topology does not change, and considering this uncertainty is left for future work.

On the other hand, we will consider uncertainty in the traffic. This refers to the fact that the flows traversing the domain are not perfectly known. Knowing exactly the volumes of these flows, which we shall call Origin Destination traffic flows, requires a measurement infrastructure that is not always present, or could be expensive to implement. Techniques based on flow-level measurements, like Netflow [20] are very expensive for routers in terms of computational cost, while their sample-based version can lead to errors in traffic volume estimation. Techniques based on SNMP data considerably reduce the CPU load on routers. In that case, the measured data consists of volumes of traffic traversing the different links of the network. In order to estimate the Origin Destination traffic volumes an ill-posed linear inverse problem has to be solved. Several methods exist in the literature for doing so, for example [21, 22, 23]. Moreover, traffic uncertainty is not only related to the complexity on measuring the Origin Destination flows, but also to the fact that traffic may change rather frequently. There can be several reasons for these changes, for instance, external routing modification, the presence of unexpected events such as network equipment failures outside the domain, large-volume network attacks or flash crowd occurrences [24].

In summary, we aim at finding a valid end-to-end QoS metric. Thus, two approaches could compete. Either we follow a dynamic approach, in which network state is continuously monitored and the metric value is updated, or we use a robust approach, in which a bound for the metric is provided. Reactive approaches make it possible to tightly follow the variations of the traffic but they require a monitoring infrastructure to be present and some sophisticated algorithms to process the measurement data. Moreover, reactive approaches are able to detect variations in the traffic demand such as abrupt changes but they are not able to forecast them [25]. On the contrary, proactive mechanisms provide pessimistic values of QoS metrics but they are able to provide metrics values which should hold for a given period of time since in that case uncertainty is taken proactively into account.

In this work we employ the proactive approach and consider the situation where traffic variation is the principal cause of delay variation. Thus, we shall focus on the computation of a bound for the end-to-end delay of traversing an AS through a particular path as a function of the AS parameters we

mentioned before: the routing configuration, the traffic demands and the traffic injected through the new tunnel. We assume that the topology and the routing configuration are fixed. However, we consider that traffic is non-static, and that it is contained in a so-called uncertainty set [26]. The question of how to choose this set is discussed later in the paper.

In this context, we provide an exact method an approximate solution for solving the problem, which renders a solution arbitrarily close to the exact solution and lower computation complexity. These solutions do not require any complex monitoring infrastructure to be deployed. The solutions are tested using topology and data from real networks.

The obtained value can be afterwards used to advertised in the context of ASs path Selection, since it is a QoS parameter bound and does not introduce confidentiality vulnerabilities. The latter refers to the fact that no topology information is delivered, just the delay of traversing the AS, where the AS is seen from the outside as a black box.

The remainder of this paper is organized as follows. Section 2 introduces the assumptions and notations and formally states the problem. In Section 3 we show an exact solution to the problem and evaluate it through simulations. Section 4 presents an approximate solution with lower computational complexity than the exact one. Section 5 shows numerical results. Finally, conclusions and guidelines for future work are given in Section 6.

2 Problem Statement

In this section we formally present the problem of finding the maximum end-to-end delay experienced by a bounded amount of traffic traversing an AS through a particular path. As mentioned before, we will consider that traffic varies within an uncertainty set. First, let us introduce the notations that are going to be used throughout the paper and state some assumptions.

2.1 Assumptions and Notations

The network is compounded of n nodes and of a set L of links, $L = \{l_1 \dots l_{|L|}\}$, where the notation $|\cdot|$ refers to the cardinality of the set. Traffic demands will be represented by the so-called traffic matrix $TM = \{tm_{i,j}\}$, where $tm_{i,j}$ is the mean amount of traffic from node i to node j. We shall use as well the term Origin Destination (OD) flows to refer to them. We reorder every

traffic demand and rewrite the OD flows ($tm_{i,j}$) in vector form as t, $t = \{t_k\}$, $k = 1 \dots n(n-1)$. The amount of traffic coming from the interdomain injected into the new tunnel will be u.

The link load $Y = \{y_i\}_{i \in L}$ is a vector containing in the i-th entry the load on link l_i without considering u. With these definitions we can see that $Y = R.t$ where R, a $|L| \times m$ matrix ($m = n(n-1)$), is the routing matrix, which means that $R_{i,j} = 1$ if flow j traverses link i, and 0 otherwise.

The flow that carries u will traverse the AS from an origin to a destination node following a certain path. We will call this path P. We will equally refer to the set of links that belong to that path as P, in this case P is a subset of L.

The mean link delay is approximated by the M/M/1 model, that is to say $D_l = \frac{K}{c_l - y_l}$, where c_l is the capacity of the link l and K the mean packet size. We then obtain the delay of a path as the sum of the mean delay of the links it traverses:

$$Delay_P = \sum_{l \in P} \frac{K}{c_l - y_l}. \tag{1}$$

The propagation delay may be ignored in our formulation since it does not change with the load and may be added as a constant later on. Moreover, the M/M/1 model is used for illustrating the procedures towards a solution. In fact, any convex function may be used instead. For instance, the interested reader should consult [27] for a method to obtain a good convex approximation of the delay function based on measurements. The same procedures explained in this paper should be then repeated but with the new function. We will as well ignore the constant K in the following formulations, for the sake of notations simplification.

2.2 Traffic Demands

As mentioned above, we will not make any assumptions on the traffic matrix except that it always belongs to a certain uncertainty set. In particular we will follow the approach presented in [26] and define the uncertainty set as a polytope formed by the result of the intersection of several half-spaces. Consequently, all constraints can be written as $At \leq b$, where A is a certain matrix that can be defined according to different models, and b is a given bound.

One such polytope was considered in the so-called Hose Model [28] in

the context of VPN services specification. This model establishes bounds in the ingress and egress points of a network. It is suitable for the case of VPN, where the ingress and egress values are easily known, but no detailed information regarding the network is available. However, in the context of interdomain path selection, we would like to have a smaller polytope, which is obtained with more detailed information, which would allow us to have a tighter bound.

An alternative is the *Links Capacity Model*. This model results from the application of bounds on the total traffic traversing the different links of the network. Its definition can vary from a simple static one, imposing the physical constraints, i.e. links capacity, to a more dynamic one, allowing the constraints to be obtained from historical metrics. In the latter, the constraints can be written as $R^h t \leq b$, where $b = \{b_i\}$ is the vector of an historical link load and R^h is the routing matrix at the moment when the measurements were taken. This approach is used for example in [29] where a polyhedral definition of the traffic matrix is preferred to its estimation because of non stationarity artifacts and estimation errors.

The Links Capacity Model with historical bounds, for instance considering the maximum observed link load, provides more detailed information than the Hose Model, along with dynamism, while it is still simple to obtain. The polytope can be frequently updated but does not require complexity for its computation.

Yet another alternative for computing tighter polytopes are prediction based mechanisms. In this case the polytope is defined through imposing bounds on the value of traffic demands which are based on traffic prediction. The prediction of future demands is based on past observations. For example artificial intelligence methods such as neural networks or time series analysis can be used in order to forecast the future values of the traffic demand; see for example [30] for prediction based on a seasonal ARIMA model. These mechanisms provide a more dynamic polytope, which must be updated according to predictions but involves more complexity. The result is a tighter polytope that provides, in turn, a tighter bound.

The choice of the model for defining the polytope involves a trade-off between complexity and tightness of the bound. As we have shown above, simpler approaches could be used providing looser bounds, or more complex ones, needing in addition to be updated frequently, to provide tighter bounds. In the remainder of this work we shall use the Links Capacity Model, though the solutions provided are still valid for any other model. We shall consider

historical maximums for the bounds, thus measurements have to be carried out. These measurements can be performed using SNMP, which is a widely deployed protocol. Since the value needed is just the overall interface traffic volume, we can safely assume that these values are going to be available on any AS.

2.3 Mathematical Formulation

For the path traversed by the new tunnel the maximum link delay is going to be computed allowing the flows t to vary within a polytope. Therefore, we will work with a maximization problem with linear constraints. In order to have a more compact notation of the problem we shall define the m-dimensional column vector w_l, $l \in P$, as $w_l = \{w_{l,i}\} = R_{l,i}/c_l$.

The optimization problem is described by (2), where A and b define the polytope.

$$\begin{aligned} \max_{t} \quad & \sum_{l \in P} 1/c_l \frac{1}{1 - w_l^T t - u/c_l} \\ \text{s.t.} \quad & At - b \leq 0. \end{aligned} \tag{2}$$

Please note that if some additional linear constraints must be taken into account they can be integrated in the definition of the polytope $At \leq b$. Example of such constraints can be $w_l^T t + u/c_l < 1$, for $l \in P$, which simply states that there should be enough link capacity in order to accommodate all the traffic, including the new tunnel.

We can see that the objective function in the maximization problem (2) is not a concave function, consequently, the problem is not a convex one. On the contrary, the problem is the maximization of a convex function over a polytope. This is a very difficult problem, all the more so since the objective function is not strictly convex (as shown in Appendix A).

Intuitively we can see that the reason why the function is not strictly convex is due to the difference between the number of links and the number of OD flows. Indeed, while the number of links grows linearly with the number of nodes in the network, the number of OD flows squares with the number of nodes in the network. This means that for different values of the vector t, the objective function of Problem (2) can have the same value, while its gradient remains always non-negative. In the following section we reformulate the problem and show a possible way to find its solution.

3 Finding the Exact Solution

3.1 Formulation

We now state the problem in a different way and show a method for finding the exact solution. We aim at formulating the problem in such a way that the objective function is strictly convex and the dimension of the problem is reduced. For doing so we shall decompose the vector t over a particular basis of $\mathbb{R}^m$.

The procedure consists in decomposing the vector t over the vectors w_l, $l \in P$, and their orthogonal complement. We define the matrix W_1 as an m by $|P|$ matrix, whose columns are the vectors w_l, with $l \in P$, and W_2, an m by $m - |P|$ full rank matrix such that it verifies

$$W_1^T W_2 = 0. \tag{3}$$

In other words, the columns of W_2 form a basis of the space orthogonal to the one spanned by the columns of W_1.

Provided that the columns of W_1 are as well linearly independent, the columns of the matrix W defined after W_1 and W_2 as

$$W = [W_1 W_2] = [w_1, \ldots, w_l, \ldots, w_{|P|}, \ldots w_m] \tag{4}$$

represent a basis of $\mathbb{R}^m$.

We shall decompose the vector t over the defined basis using the auxiliary variables $x \in \mathbb{R}^{|P|}$ and $h \in \mathbb{R}^{m-|P|}$ as

$$t = W_1 x + W_2 h. \tag{5}$$

By multiplying both sides of Equation (5) by w_l^T, and using Equation (3) we obtain

$$w_l^T t = w_l^T W_1 x = v_l^T x, \tag{6}$$

where we have set $v_l^T = w_l^T W_1$, for all $l \in P$. Note that both v_l and x are column vectors of dimension $|P|$.

Equation (6) will directly lead us to rewriting the objective function of Problem (2) as a function of x. We shall now redefine the polytope by writing it in the basis W. For doing so the change of variables defined by Equation (5) needs to be done in the constraints of Problem (2). This leads to defining a new matrix denoted D and computed as AW. The polytope over the new basis can be compactly written as $D[x^T \ h^T]^T \leq b$.

All in all, Problem (2) can be rewritten in the form of Problem (7). Please note that the objective function depends only on the variable x.

$$\begin{aligned} \max_{x} \quad & \sum_{l \in P} 1/c_l \frac{1}{1 - v_l^T x - u/c_l} \\ \text{s.t} \quad & D \begin{pmatrix} x \\ h \end{pmatrix} \leq b. \end{aligned} \tag{7}$$

Let us call the objective function of Problem (7) as $J(x)$ and the new polytope as V (i.e. $V = \{[x^T \ h^T]^T \in \mathbb{R}^m : D[x^T \ h^T]^T \leq b\}$). Let us as well define the polytope V_x as

$$V_x = \{x \in \mathbb{R}^{|P|} \mid \exists \ h \in \mathbb{R}^{m-|P|} : D[x^T \ h^T]^T \leq b\}. \tag{8}$$

Let $\mathcal{W}_1 = span\{w_1 \dots w_{|P|}\}$, where span refers to the set of all linear combinations of vectors $w_1 \dots w_{|P|}$. Clearly V_x is the projection of V onto $\mathcal{W}_1$.

Since V is a convex polytope by definition, it is easy to check that V_x is also a convex polytope. More precisely, V_x is the convex hull of the projection of the extreme points of V onto $\mathcal{W}_1$ [31].

Then, since $J(x)$ does not depend on h, Problem (7) can be represented in the space $\mathcal{W}_1$ as follows:

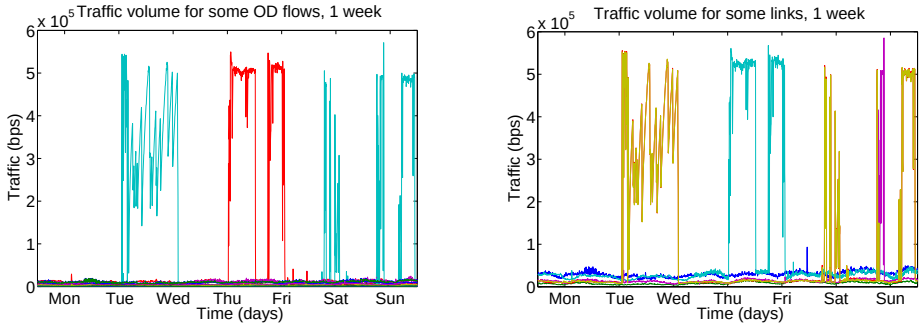
$$\begin{aligned} \max_{x} \quad & J(x) \\ \text{s.t.} \quad & x \in V_x. \end{aligned} \tag{9}$$

It is proven in Appendix B that $J(x)$ is strictly convex in V_x and that the solution of Problem (9) is attained at an extreme point of the polytope V_x.

Problem (9) allows us to work with a strictly convex function, and to reduce the dimension of the feasible region, in some cases, considerably. In order to find the solution, we need to be able to perform the projection of a polytope, and afterwards enumerate its extreme points. Methods for doing so are available (see for instance [32]), although these can be computationally expensive tasks. In the following subsection we explore this solution by performing simulations in a real topology.

3.2 Simulations

In order to assess the results of the proposed method we will use the Abilene network, whose topology, historical traffic demands and routing matrix are



(a) Traffic volume per OD flow.

(b) Link load.

Figure 2: Example of traffic variation in the Abilene network, one week of traffic.

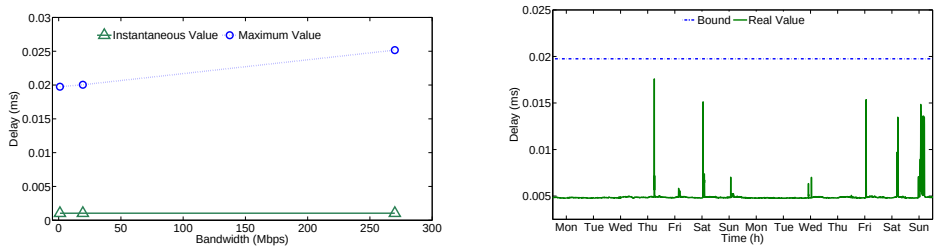
available from [33]. The Abilene Network consists of 30 internal links and 12 routers (all exchanging traffic among them).

For defining the polytope linear constraints were set using the links capacity model with a maximum in the link utilization equal to the maximum achieved by historical data. We actually did a scaling of these data in order to obtain a more interesting case, since historically link utilizations in the Abilene Network have been quite small.

Figure 2 shows a traffic trace of Abilene Network. This example shows how the traffic matrix is prone to sudden traffic variations. Figure 2(a) shows the traffic for some OD flows corresponding to 2016 consecutive measurements (where each color corresponds to one OD flow), while Fig. 2(b) shows the link load (each color corresponds to the load of a certain link).

In order to test the method in different scenarios, we compute the solution for four paths between a pair of Origin and Destination nodes and three different amount of bandwidth for the interdomain request. The paths were computed using a Shortest Path algorithm, and keeping the four shortest paths. Please note that the way of selecting the paths could have been a different one. This choice and its impact on the delay are out of the scope of the present paper. The values for the bandwidth of the interdomain demands (i.e. u) are 1 Mbps, 19.4Mbps and 270Mbps. These values could correspond, for instance, to a VoIP service, a broadcast quality HDTV service and a VPN service, respectively.

For each path we will compute the maximum delay suffered by a flow



(a) Comparison of the maximum delay value and an instantaneous one for one path.

(b) Comparison of the real delay values and a computed bound for one path and one bandwidth demand.

Figure 3: Simulations on the Abilene network.

traversing it and carrying each one of these services at a time. All results were computed on a recent machine with good computational power (two processors Intel Xeon X5660 2.80GHz, 24GB of RAM). For computing the polytope projection and enumerating its extreme points we used the MPT library [34].

In Fig.3(a) a comparison of the maximum delay value for the fourth shortest path and the real mean delay of the same path at a certain instant is shown. For this particular case the maximum delay value is approximately 3 times more than the instantaneous one. This highlights the need of announcing a value that the domain is able to assure, rather than a measured one. Figure 3(b) shows the same bound, compared to several delay values occurred during the two weeks after computing the bound. Again, this shows that the delay can vary considerably and considering a value obtained in a given moment could not be safe. The bound in this case is quite loose, it should be noticed that it could be tightener by including more information in the description of the polytope, for instance, considering at the same time the Hose model, as introduced in Subsection 2.2.

The time consumed to perform the computation of all demands for one path varied between 4 and 38 minutes, which for a moderately sized network is rather high. In fact, even if in several topologies we were able to find the exact solution through these means, it is still an open question whether there exists an algorithm for enumerating all extreme points of a polytope of an arbitrary dimension in polynomial running time [35]. In the following section, we present an approximate method that can be used as an alternative to the

previous one when its computational time becomes excessive.

4 Finding an Approximate Solution

In Section 3 we have presented a method that allows to find the exact solution. Nevertheless, we have pointed out that its complexity remains open. In this sense, we now present a method that provides an approximate solution to Problem (2), while reducing the computational time. More precisely, we present a numerical method based on the approximation of the objective function by a piecewise linear function. This method provides a value that is arbitrarily close to the exact solution (up to some controlled error).

Let us introduce the method with a detailed description of the procedure to obtain it. First of all, we transform each link's delay function, $\frac{K}{c_l}\frac{1}{1-y_l/c_l-u/c_l}$, into a piecewise linear function over y_l. For this, we partition each function's domain into η_l subintervals and approximate the function in each subinterval by its first order Taylor polynomial. We shall note the subintervals of link l as $\Delta_{l,j}$, $j = 1 \dots \eta_l$.

Secondly, we obtain the delay of the path, as before, by summing up the delay on each link belonging to it. Therefore, we obtain a maximization problem similar to (2) but now with a piecewise linear objective function. Let us utilize the indicator function, defined as

$$\mathbb{1}_{\Delta}(x) = \begin{cases} 1 & \text{if } x \in \Delta \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

Then, the new problem can be seen in (11), where α and β are taken from the Taylor polynomial of the original function.

$$\begin{aligned} \max_{t} \quad & \sum_{l \in P} \sum_{j=1}^{\eta_l} \left(\alpha_{l,j} w_l^T t + \beta_{l,j}\right) \mathbb{1}_{\Delta_{l,j}}(w_l^T t) \\ \text{s.t.} \quad & At - b \leq 0. \end{aligned} \tag{11}$$

The next step is to redefine Problem (11) in order eliminate the indicator function and to obtain a linear objective function. To do so, we decompose Problem (11) into $\prod_{l \in P} \eta_l$ problems, each of them having a linear function

as objective one. This linear function stems from the consideration of one of the linear functions that compound each link delay's approximation, and summing them up. Let us now use the index $j(l)$, $j(l) = 1 \dots \eta_l$ for all $l \in P$, to denote the linear function chosen for link l, corresponding to subinterval $\Delta_{l,j}$. In order to consider each linear function only on the corresponding domain we introduce new constraints to the problem. That is to say, the solution has to be restricted to belong to the original polytope and at the same time to the set $\{t \in \mathbb{R}^m | w_l^T t \in \Delta_{l,j}\}$. It can be readily proved that this set is equivalent to imposing restrictions on the load on each link. Thus, it is itself a polytope. We represent the intersection of the original polytope and the new one, which is also a polytope, in the matrix form as the set $\{t \in \mathbb{R}^m | A^* t \leq b^*\}$, where the matrix A^* and the vector b^* define the intersection polytope.

Finally, in order to have a problem equivalent to Problem (11) we consider all combinations of linear functions for each link, find the maximum over t for each combination and keep the combination which leads to the greatest value of the objective.

The mathematical formulation of the equivalent problem can be seen in (12), where the maximum on $j(1), j(2), \dots, j(|P|)$ means that we consider the maximum obtained when we let each value $j(l)$ vary between 1 and η_l.

$$\max_{j(1),j(2)\dots j(|P|)} \begin{cases} \max\limits_{t} & \sum\limits_{l \in P} \alpha_{j(l)} w_l^T t + \beta_{j(l)} \\ \text{s.t.} & A^* t - b^* \leq 0. \end{cases} \tag{12}$$

As we have claimed above, this method leads to a solution that is arbitrary close to the exact solution of the original problem (Problem (2)). The proof of this statement is shown in Appendix C.

Problem (12) can be solved computationally by performing a loop of $\prod_{l \in P} \eta_l$ iterations. Please note that the problem solved on each iteration is a linear one, which is very easy to solve.

For obtaining the partition needed to define the piecewise linear function, we propose to iteratively compute the subintervals such that within each of them the maximum difference between the approximate function and the original one is a given ϵ, at most. This constructive procedure is shown in Appendix C, and is part of the proof of achieving a solution arbitrary close to the exact solution.

In order to reduce the number of iterations, we pre-compute the maximum

value that the load can achieve at each link according to the constraints imposed by the polytope. Table 1 shows the number of subintervals needed to define the piecewise linear function, for different percentage errors and maximum link utilization (LU). This gives an idea of the complexity of the procedure. For example, for a 6-link length path, at most $4^6 \approx 4000$ linear problems need to be resolved for obtaining a result with 10% of error, thus the numerical complexity is still feasible.

5 Numerical Results

In order to assess the results of the numerical approximation method we shall first use the Abilene network as before. We will as well use the same polytope as before, so as to be able to compare results afterwards. As optimization software we use CPLEX [36]. We shall secondly perform further simulation studies on another network, so as to be able to have more information about the computational time.

We compute the maximum delay for the four paths and three interdomain demands used in Subsection 3.2, using the numerical approximation method. The results along with the exact solutions are displayed on Fig. 4, where the bars indicate the maximum error (10% in this case). Overall, the computation of each of the aforementioned values takes in mean 2.28 seconds. The maximum link utilization (imposed by the topology) was between 30% and 80%. These computational times are dramatically smaller than the ones necessary for obtaining the exact solution (approximately a 1000× decrease) while providing a very tight bound.

The previous simulations allowed us to validate the method and show that its computational time is much smaller than the one obtained through the exact method. We shall now explore this computational time when varying different parameters of the problem, namely the error, the maximum links' load and the number of links in the path.

The computational time depends on the accuracy needed by the application, which is not established a priori since it is a decision to be taken by each AS. It depends, in addition, on the maximum link utilizations allowed by the polytope along with the topology, and on the number of links compounding the path.

In order to asses the impact of the accuracy in the time consumed by the procedure we repeated the simulations allowing a maximum error of 5%

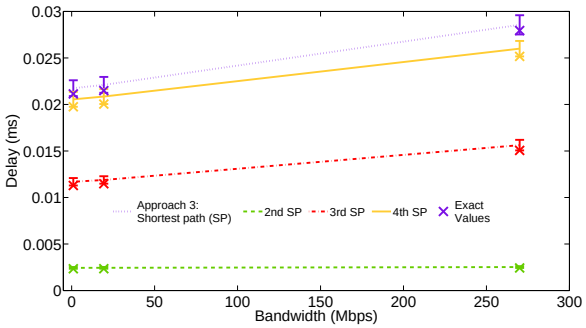


Figure 4: Approximate and Exact Solution: Maximum delay for one Origin-Destination pair for different Bandwidth demands and paths.

and of 2%. The mean time needed for computing the maximum delay over one path for one interdomain demand is of 1 and 7.9 seconds, for obtaining a solution within 5% and 2% of error respectively, which implies approximately a 90× decrease for the 2% error, with respect to the exact method.

The previous results were obtained using historical traffic demands over the Abilene Network. This implies that the maximum link load, imposed by the polytope, is between 30 and 80%, as mentioned before. We now present further simulation results using synthetic data to define the polytope, so as to obtain results on a scenario with higher maximum link utilizations.

Figure 5 shows the ratio of the time consumed for computing the delay bound through the Approximate Method to the time consumed by the Exact Method. The bound was computed for one interdomain demand of 19.4 Mbps for a polytope imposing maximum link utilizations between 80 an 90% and three different values of errors (i.e. ϵ). This was repeated for the same four different paths presented above. Results of these simulations show that the time consumed by the Approximate Method is much less than the one consumed by the Exact Method. In the worst case, that is, $\epsilon = 2\%$ and the 4th shortest path, the time consumed by the approximate method is of 19 minutes while the time consumed by the Exact Method is of 38 minutes.

Finally, we shall explore the influence on the computational time of the number of links in the path. For doing so, let as utilize a larger network, the GÉANT network [37], which is composed of 23 nodes and 74 links. Figure 6 shows the computational time for one interdomain demand of 19.4 Mbps

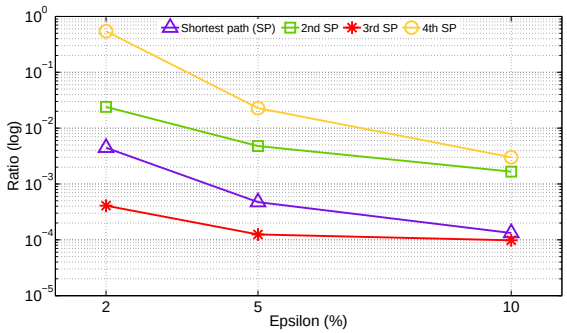


Figure 5: Ratio of the time consumed by the Approximate Method to the Exact Method: one interdomain demand for different allowed error and maximum link utilization between 80 and 90%. Bound computed for four different paths over the Abilene network.

from an origin to a destination node, through different paths. Results are presented for a link utilization between 20% and 90% and different values of the allowed error. Is it worth clarifying that the topology does not allow to have high link utilizations in all links at the same time, since there are a number of bottleneck links on it. Results show that the computational time is not very sensitive to the path-length. For the case of a 11-link path, which greatly exceeds the maximum path length on a domain, is of approximately 7 minutes, providing a value within the 2% of error.

The Approximate Method was shown, through extensive simulation studies, to consume low computational times in most of the cases. The methods proposed on this work are conceived to be used by each AS in order to obtain a value of a metric to announce in the process of Interdomain Path Selection. The announced bound is supposed to hold for a long period of time, for instance, several hours. In this context, the time consumed by the Approximate Method is considered totally acceptable. However, we have not focused our work on optimizing this computational time, which could be, for instance, diminished through parallelizing the code, since its nature allows it (it solves an optimization problem over several independent feasible regions).

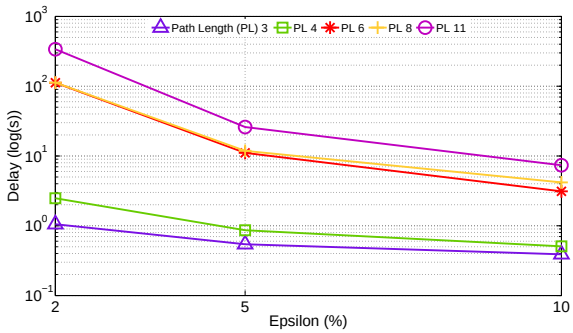


Figure 6: Time consumed by the Approximate Method: one interdomain demand for different allowed error and maximum link utilization between 20 and 90%. Bound computed for different paths over the GÉANT network.

6 Conclusion and Future Work

We have presented a means of computing a bound on the end-to-end delay. The method takes into account the uncertainties in the traffic traversing the AS, which have been modeled as a polytope. Therefore, it is a value that the AS can guarantee for a certain period of time. The problem was mathematically stated and different solutions were provided. A method for finding the exact solution was given, and an alternative approximate method was proposed, as a remedy for the high computational cost of the former. Such approximate method renders rather than a value of the delay, an interval to where the real value of the maximum delay is guaranteed to belong. The latter was theoretically proven and numerically validated, by comparing the results to the real maximum. Both methods were tested on real networks using measurement and synthetic data. The Approximate method was shown, through simulations, to provide acceptable computational times on several scenarios. Altogether, we have proposed a method to enhance PCE-based interdomain path selection mechanisms, which can be implemented with low computation complexity and little monitoring infrastructure.

As future work, we shall explore the case of having uncertainty on the AS topology in addition to traffic uncertainty. For instance, taking into account the case of link or node failures, and being able to provide even in those cases a tight end-to-end delay bound. We shall as well explore the possibility of building a delay curve as a function of ingress traffic. If the delay can

be advertised as a function of ingress traffic, this would allow to take more sophisticated routing decisions, leading to lower end-to-end delay values. In addition, in the scenario considered in this paper where ASs collaborate among them, the problem of distributing all flows among the ASs involved for achieving social welfare at the time they have QoS requirements becomes an interesting one and raises new questions. Studying such scenario using the present approach as a building block, is a challenging work which we shall address in the future.

Acknowledgements

The research leading to these results has received funding from the European Community Seventh Framework Programme (FP7/2007-2013) under grant agreement n°248567 for the ETICS Project. Further information is available at `www.ict-etics.eu`. This work was also partially funded by the Uruguayan Agency for Research and Innovation (ANII) under grant PR-POS-2008-003.

A Convexity of the Objective Function

In this section we prove that the objective function of Problem (2) is a convex function but not a strictly convex one. We use the notations introduced in Section 2.1. In addition, we will denote the objective function of Problem (2) as $f(t)$ and the feasible region of such problem as S.

We are interested in finding out if $f(t)$ is a convex function over S, thus we shall explore if the following inequality holds [38]

$$f(t_1) \geq f(t_2) + \nabla f(t_2)^T (t_1 - t_2),\ t_1, t_2 \in S. \tag{13}$$

Applying the definition of f to Equation (13) we obtain the following inequality

$$\sum_{l \in P} \frac{1/c_l}{1 - w_l^T t_1 - u/c_l} \geq \sum_{l \in P} \frac{1/c_l}{1 - w_l^T t_2 - u/c_l} + \sum_{l \in P} \frac{1/c_l w_l^T (t_1 - t_2)}{(1 - w_l^T t_2 - u/c_l)^2},\ t_1, t_2 \in S. \tag{14}$$

Let us now define $g_l(t)$, an auxiliary function in order to simplify the notations, as

$$g_l(t) = 1 - w_l^T t - u/c_l,\ t \in S. \tag{15}$$

Substituting the latter definition in Equation (14) and performing some regular math operations we obtain the following inequality

$$\sum_{l \in P} \frac{(g_l(t_2) - g_l(t_1))^2}{g_l(t_1) g_l(t_2)^2} \geq 0, \ t_1, t_2 \in S. \tag{16}$$

Please note that each term on Inequality (16) is either zero or greater than zero for all t_1, $t_2 \in S$. Therefore, the function f is convex over S. It remains to show if the function is strictly convex or not.

Therefore, we need to find out if there exist t_1 and $t_2 \in S$ such that $g_l(t_2) - g_l(t_1)$ is equal to zero for all $l \in P$. The latter is equivalent to showing that there exist t_1 and $t_2 \in S$ such that $< w_l, t_2 - t_1 >$, the scalar product of vector w_l and $(t_2 - t_1)$, is equal to zero for all $l \in P$, that is to say, having all vectors w_l, $l \in P$ orthogonal to the vector $(t_2 - t_1)$. Since the vectors w_l do not form a basis of $\mathbb{R}^m$ it is possible to find t_1 and $t_2 \in S$ such that their difference is orthogonal to all vectors w_l, $l \in P$. This finishes the proof, which concludes that f is a convex function, but not a strictly convex one.

B Convexity of $J(x)$ and Characterization of the Optimal Point

In this appendix we shall prove that the function $J(x)$ defined in Section 3 is indeed a strictly convex function over V_x and that the solution to Problem (9) is an extreme point of the polytope V_x. We shall use the notations introduced in Sections 2.1 and 3.

Let us define $\lambda_l(x)$ as

$$\lambda_l(x) = (1 - v_l^T x - u/c_l)^{-2}, \ \forall l \in P \tag{17}$$

and the matrix Λ as

$$\Lambda(x) = diag(\lambda_1, \ldots, \lambda_{|P|}). \tag{18}$$

For all $x \in V_x$ and $l \in \{1 \ldots |P|\}$, $\lambda_l(x) > 0$. Thus, $\Lambda(x)$ is a positive-definite matrix[1].

[1] A $n \times n$ real symmetric matrix M is positive-definite if $z^T M z > 0$ for all non-zero vectors z, $z \in \mathbb{R}^n$.

In addition, we can check that $[v_1 \dots v_{|P|}] = W_1^T W_1$ is also a positive-definite matrix. Thus, the Hessian of $J(x)$, which is

$$\nabla^2 J(x) = (W_1^T W_1)\Lambda(x)(W_1^T W_1) \tag{19}$$

is as well a positive-definite matrix, which concludes the proof.

We are now able to show that the solution to Problem (9) is attained at an extreme point of V_x.

Indeed, we can prove by contradiction that the maximum of $J(x)$ over V_x must be reached at an extreme point of V_x. Since J is a strictly convex function, inequality (20) holds [38].

$$J(\Phi) > J(\theta) + \nabla J(\theta)^T(\Phi - \theta),\ \forall\ \theta, \Phi\ \ \in V_x. \tag{20}$$

Now, let $\bar{\theta} \in V_x$ be an optimal point of Problem (7). Therefore, $\bar{\theta}$ is a strict maximum, since J is strictly convex, and, for all $\Phi \in V_x \setminus \{\bar{\theta}\}$, we must have:

$$J(\Phi) - J(\bar{\theta}) < 0. \tag{21}$$

Together with inequality (20), we get

$$\nabla J(\bar{\theta})^T(\Phi - \bar{\theta}) < 0,\ \forall \Phi \in V_x \setminus \{\bar{\theta}\}. \tag{22}$$

By contradiction we suppose that $\bar{\theta}$ is not an extreme point of V_x. Then there exists $\mu \in \mathbb{R}^{|P|}$ such that $||\mu|| > 0$ and $\bar{\theta} + \mu$, $\bar{\theta} - \mu \in V_x$. By letting $\Phi = \bar{\theta} - \mu$ and $\Phi = \bar{\theta} - \mu$ at a time, we would get:

$$\nabla J(\bar{\theta})^T \mu < 0 \text{ and } -\nabla J(\bar{\theta})^T \mu < 0, \tag{23}$$

which is not possible.

This allows us to conclude that $\bar{\theta}$ can not be a non extremal point of V_x, which finalizes the proof.

C Validation of the Approximate Method

In this Section we prove that the Approximate Method presented in Section 4 reaches the exact solution of Problem (2), up to some controlled error.

We remind the reader that we start from a problem like Problem (2), we then partition the feasible region into several sub-regions and approximate the objective function of Problem (2) as a linear function within each of the

sub-regions. This problem can be seen in Problem (12). We then keep the maximum of the output of all the possible problems of maximization over t in Problem (12).

Let us call the original function, defined in Problem (2), as $f(t)$ and the piecewise linear approximation of f as $\tilde{f}$. Let $\tilde{t_f}$ and t_f be the values at which the maximum of $\tilde{f}$ and f are attained respectively. Let us as well note the feasible region of Problem (2) as S.

We set the hypothesis that for a given real positive ϵ, the approximation of f can be made such that the difference between f and $\tilde{f}$ is bounded by ϵ. That is to say that

$$f(t) - \tilde{f}(t) \leq \epsilon \ \forall t \in S. \tag{24}$$

Under the conditions of Equation (24) and with the definitions of $\tilde{t_f}$ and t_f provided above we can prove that

$$f(t_f) - f(\tilde{t_f}) \leq \epsilon. \tag{25}$$

Let us first prove that Equation (24) holds for the case of the M/M/1 model mean delay function. Please note that for other functions this is an hypothesis to be checked before applying the algorithm. We provide a constructive proof showed in the following. For the sake of simplicity on the notations we will not include u in the formulation, but the whole procedure can be reproduced in an analogous way considering u.

Let $e(y) = \frac{1}{1-y}$ be such that $f(t) = \sum_{l \in P} 1/c_l e(w_l^T t)$, $t \in S$. Let us note the partition of the domain of e over y ($[0, 1)$) as the set of subsets Δ_i where

$$\Delta = \{\Delta_i : i = 1 \dots \eta\}. \tag{26}$$

Let $\tilde{e}$ be the piecewise linear approximation of e over each one of the subsets defined in (26), such that the following inequality holds

$$|e(y) - \tilde{e}(y)| \leq \frac{\epsilon}{\sum_{l \in P} 1/c_l} = \delta \ \forall y \in \Delta. \tag{27}$$

This will ensure that Equation (24) holds since $\tilde{f}(t) = \sum_{l \in P} 1/c_l \tilde{e}(w_l^T t)$.

We shall consider the graphic displayed on Fig. 7. Let us note Δ_i as $\Delta_i = [y_{i-1}, y_i]$, with $y_0 = 0$. We define $z_i \in \Delta_i$ as the linearization point of function e in Δ_i. Let us define ϵ_{i-1} and ϵ_i as the difference between e and $\tilde{e}$ at each y_{i-1} and y_i respectively. That is to say

$$\epsilon_{i-1} = e(y_{i-1}) - \tilde{e}(y_{i-1}) \text{ and } \epsilon_i = e(y_i) - \tilde{e}(y_i), i = 1 \dots \eta. \tag{28}$$

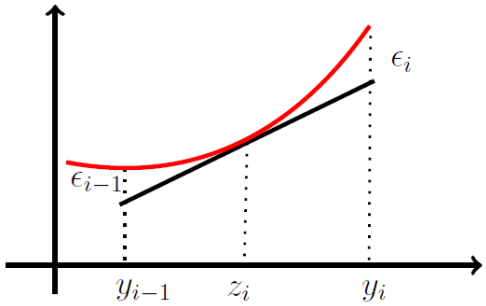


Figure 7: Difference between function e and its approximation $\tilde{e}$.

It is not difficult to see that the maximum of the difference between e and $\tilde{e}$ will be attained at either y_{i-1} or y_i, $i \in 1 \dots \eta$.

Given y_{i-1} fixed, let z_i increase from y_{i-1}. We define y_i such that $\epsilon_i = \delta$. As z_i increases, y_i and ϵ_{i-1} increase. Therefore, the maximum subinterval size under the constraint $\epsilon_{i-1}, \epsilon_i \leq \delta$ is achieved when

$$\epsilon_{i-1} = \epsilon_i = \delta. \tag{29}$$

Therefore, given y_{i-1} and δ, we can find a value z_i imposing that

$$e(y_{i-1}) - \tilde{e}(y_{i-1}) = \delta. \tag{30}$$

Once z_i is known, we can compute y_i by imposing

$$e(y_i) - \tilde{e}(y_i) = \delta. \tag{31}$$

All in all, it appears that e can be approximated over its domain by means of a piecewise linear function. Thus, $f(t) = \sum_{l \in P} 1/c_l e(w_l^T t)$ can be approximated by means of a sum of piecewise linear functions, and this approximation, which we note as $\tilde{f}(t)$, is such that Equation (24) holds for all $t \in S$.

We have provided a constructive proof of Equation (24). We are now able to show that (25) holds. Indeed, let again $\tilde{t_f}$ and t_f be the values at which the maximum of $\tilde{f}$ and f are attained respectively. The following inequalities are obtained straightforwardly from the definition of the maximum

$$\tilde{f}(\tilde{t_f}) \geq \tilde{f}(t_f) \tag{32}$$

$$f(t_f) \geq f(\tilde{t_f}). \tag{33}$$

We are interested in finding a bound to the difference $f(t_f) - f(\tilde{t_f})$ which can be rewritten as in the following equation

$$f(t_f) - f(\tilde{t_f}) = \overbrace{\tilde{f}(\tilde{t_f}) - f(\tilde{t_f})}^{\leq 0} + \overbrace{\tilde{f}(t_f) - \tilde{f}(\tilde{t_f})}^{\leq 0} + \overbrace{f(t_f) - \tilde{f}(t_f)}^{\leq \epsilon}. \quad (34)$$

Equation (34) immediately leads to the inequality $f(t_f) - f(\tilde{t_f}) \leq \epsilon$, which completes the proof. Please note that in Equation (34) we have used the fact that $f(t)$ is greater than $\tilde{f}(t)$ for all $t \in S$, which is true since $\tilde{f}$ is the piecewise linear approximations of a convex function.

References

[1] R. Zhang, J. P. Vasseur, RFC 4216 - MPLS Inter-Autonomous System (AS) Traffic Engineering (TE) Requirements, Internet Engineering Task Force. (November 2005).

[2] A. Farrel, J. P. Vasseur, A. Ayyangar, RFC 4726 - A Framework for Inter-Domain Multiprotocol Label Switch, Internet Engineering Task Force. (2006).

[3] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow, RFC 3209 - RSVP-TE: Extensions to RSVP for LSP Tunnels, Internet Engineering Task Force. (December 2001).

[4] Path computation element (PCE) IETF working group, http://www.ietf.org/html.charters/pce-charter.html (Last Visited June 2011).

[5] D. King, A. Farrel, The Application of the Path Computation Element Architecture to the Determination of a Sequence of Domains in MPLS and GMPLS (July 2009).

[6] H. Pouyllau, R. Douville, N. B. Djarallah, N. Le Sauze, Economic and technical propositions for inter-domain services, in: Bell Labs Tech. J., Vol. 14, Bell Labs Villarceaux, France, 2009, pp. 185–202.

[7] G. Bertrand, G. Texier, Ad-hoc recursive PCE based inter-domain path computation (ARPC) methods, in: Fifth International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs), 2008.

[8] C. Pelsser, O. Bonaventure, Path Selection Techniques to Establish Constrained Interdomain MPLS LSPs, in: Networking 2006, Vol. 3976 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2006, Ch. 18, pp. 209–220–220.

[9] B. Quoitin, C. Pelsser, L. Swinnen, O. Bonaventure, S. Uhlig, Interdomain traffic engineering with BGP, in: Communications Magazine, IEEE, Vol. 41, 2003, pp. 122–128.

[10] M. Yannuzzi, A. Fonte, X. Masip-Bruin, E. Monteiro, S. Sanchez-Lopez, J. Domingo-Pascual, A self-adaptive QoS routing framework for multihomed stub autonomous systems, in: Proceedings of Eunice, IFIP. Madrid, Spain, 2004, pp. 241–247.

[11] S. R. Lima, P. Carvalho, V. Freitas, Toward Scalable Management of Multiple Service Levels in IP Networks, in: Proceedings of the Fourth European Conference on Universal Multiservice Networks, IEEE Computer Society, Washington, DC, USA, 2007, pp. 223–232.

[12] S. Niccolini, M. Molina, F. Raspall, S. Tartarelli, Design and Implementation of a One Way Delay Passive Measurement System, in: Network Operations and Management Symposium, 2004. IEEE/IFIP, Vol. 1, 2004, pp. 469–482.

[13] A. Hernandez, E. Magana, One-way Delay Measurement and Characterization, in: Networking and Services, 2007. ICNS. Third International Conference on, 2007, p. 114.

[14] J. Corral, G. Texier, L. Toutain, End-to-End Active Measurement Architecture in IP Networks (SATURNE), in: in IP networks (SATURNE), in PAM2003 - A workshop on Passive and Active Measurements, 2003, pp. 3–4.

[15] S. Shalunov, B. Teitelbaum, A. Karp, J. Boote, M. Zekauskas, RFC 4656 - A One-way Active Measurement Protocol (OWAMP), Internet Engineering Task Force. (September 2006).

[16] L. De Vito, S. Rapuano, L. Tomaciello, One-Way Delay Measurement: State of the Art, in: IEEE Transactions on Instrumentation and Measurement, Vol. 57, 2008, pp. 2742–2750.

[17] X. Masip-Bruin, S. Sanchez-Lopez, J. Sole-Pareta, J. Domingo-Pascual, QoS routing algorithms under inaccurate routing for bandwidth constrained applications, in: ICC '03, IEEE, 2003, pp. 1743–1748.

[18] R. A. Guérin, A. Orda, QoS Routing in Networks with Inaccurate Information: Theory and Algorithms, in: Networking, IEEE/ACM Transactions on, Vol. 7, IEEE Press, Piscataway, NJ, USA, 1999, pp. 350–364.

[19] J. Levendovszky, C. Orosz, Developing Novel Statistical Bandwidths for Communication Networks with Incomplete Information, in: Experimental and Efficient Algorithms, 2005, pp. 614–617.

[20] B. Claise, RFC 3954 - Cisco Systems NetFlow Services Export Version 9, Internet Engineering Task Force. (2004).

[21] Y. Vardi, Network Tomography: Estimating Source-Destination Traffic Intensities from Link Data, Journal of the American Statistical Association 91 (433) (1996) 365–377.

[22] Y. Zhang, M. Roughan, C. Lund, D. L. Donoho, Estimating point-to-point and point-to-multipoint traffic matrices: an information-theoretic approach, Networking, IEEE/ACM Transactions on 13 (2005) 947–960.

[23] P. Casas, S. Vaton, L. Fillatre, T. Chonavel, Efficient Methods for Traffic Matrix Modeling and On-line Estimation in Large-scale IP Networks, in: 21st International Teletraffic Congress, 2009, pp. 1–8.

[24] R. Teixeira, N. Duffield, J. Rexford, M. Roughan, Traffic Matrix Reloaded: Impact of Routing Changes, in: Passive and Active Network Measurement, Vol. 3431 of Lecture Notes in Computer Science, Springer Berlin, 2005, Ch. 20, pp. 251–264.

[25] P. Casas, S. Vaton, L. Fillatre, I. Nikiforov, Optimal Volume Anomaly Detection and Isolation in Large-scale IP Networks Using Coarse-grained Measurements, Vol. 54, 2010, pp. 1750–1766.

[26] W. Ben-Ameur, H. Kerivin, Routing of Uncertain Traffic Demands, in: Optimization and Engineering, Vol. 6, 2005, pp. 283–313.

[27] F. Larroca, J. L. Rougier, Robust Regression for Minimum-delay Load-balancing, in: 21st International Teletraffic Congress, 2009, pp. 1–8.

[28] N. G. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. K. Ramakrishnan, J. E. Van der Merive, A Flexible Model for Resource Management in Virtual Private Networks, in: SIGCOMM '99, ACM, New York, USA, 1999, pp. 95–108.

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

[30] L. Fillatre, D. Marakov, S. Vaton, Forecasting Seasonal Traffic Flows, in: Workshop on QoS and Traffic Control, 2005.

[31] A. Brondsted, An Introduction to Convex Polytopes, Springer, 1982.

[32] C. N. Jones, Polyhedral Tools for Control, Ph.D. thesis, Pembroke College (2005).

[33] Y. Zhang, The Abilene Dataset, http://www.cs.utexas.edu/yzhang/research/AbileneTM/ (Last Visited June 2011).

[34] Multi-Parametric Toolbox (MPT) A tool (not only) for multi-parametric optimization, http://control.ee.ethz.ch/~mpt/ (Last Visited June 2011).

[35] L. Khachiyan, E. Boros, K. Borys, K. Elbassioni, V. Gurvich, Generating All Vertices of a Polyhedron Is Hard, Vol. 39, Springer New York, 2008, pp. 174–190.

[36] IBM ILOG CPLEX Optimizer, http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/ (Last Visited June 2011).

[37] The GÉANT network, http://www.geant.net (Last Visited June 2011).

[38] M. Minoux, Mathematical programming : theory and algorithms, John Wiley, 1986.

	ϵ 1 %	ϵ 10 %
LU 60%	5	2
LU 80%	9	3
LU 90%	12	4

Table 1: Number of subintervals needed to define the piecewise linear function.