Analysis and improvements to MATE algorithm

Miguel Griot, Gabriel Tucci, Pablo Belzarena, Santiago Remersaro
IIE, Facultad de Ingeniería, UDELAR, Uruguay. email:[mgriot,belza,gtucci]@fing.edu.uy

Abstract

This paper describes an implementation, analysis and improvements to MATE (MPLS Adaptive Traffic Engineering) algorithm [6]. MATE is an on-line load balancing algorithm. As MATE was originally thought for constant average incoming traffic, the first improvement is the usage of an adaptive update step size for time varying traffic. The second one modifies the time interval between updates to adequate itself to the traffic characteristics and calculate the amount of measurements in each interval to have a reliable statistic. The original algorithm and the new version, which will be called ***MATE-TV*** *(MATE for Time-Varying traffic), were implemented in a LINUX - MPLS network, using the 'mpls-linux' packet distributed by Source Forge [4].*

I. Introduction

THE MPLS (MultiProtocol Label Switching) [1] architecture allows to do Traffic Engineering (MPLS-TE) in IP networks [2]. The main MPLS function that allows Traffic Engineering is Explicit Routing which allows to create predefined paths (LSPs) for the arriving packets from the edge routers.

Load Balancing is one of the main research areas in MPLS-TE. It focuses on the idea of splitting the traffic of an aggregated flow between various pre-established LSPs based on some network optimality criterion. Diverse load balancing algorithms have been developed in the last few years. Some authors have been working on this topic, using an off-line approach in a MPLS network [3], based on *effective bandwidth* models.

Other authors have been working on load balancing in a MPLS network using on-line tools. With this approach, a load balancing algorithm named **MPLS Adaptive Traffic Engineering (MATE)**, which appears as an important step in the research and development of this topic, is proposed in [6]. An analysis of MATE is made in this article, some weaknesses of it are discussed and some improvements are proposed.

Firstly, the algorithm is designed for constant incoming traffic, not guaranteeing its convergence for time-varying traffic. Secondly, there is no criterion for determining the size of the time interval between the load balancing updates, although it is specified that this time interval must be bounded and an update must be made every time the delays start to grow. This paper proposes improvements to the original MATE algorithm that solve both problems.

II. MATE overview

MATE models the network as a set of L unidirectional links. The network is shared by a set of S ingress-egress (IE) pairs. Each one of these IE pairs has a set of P_s LSPs available.

Each IE pair s has an incoming rate r_s and routes x_{sp} amount of it on LSP $p \in P_s$ so that

$$\sum_{p \in P_s} x_{sp} = r_s \quad \text{for} \quad \text{all} \quad s \in S$$

Let $x_s := (x_{sp}, p \in P_s)$ be the rate vector of s and $x := (x_{sp}, p \in P_s, s \in S)$ the vector of all rates.

The flow on a link $l \in L$ has a rate x^l that is the sum of the source rates on all LSPs that go through the link l. Hence,

$$x^l = \sum_s \sum_{l \in p, p \in P_s} x_{sp}$$

There is a cost $C_l(x^l)$ associated with each link, which is assumed to continuous and convex.

MATE objective is to minimize the total cost function $C(x)$ defined as $C(x) := \sum_{l \in L} C_l(x^l)$ by optimally routing the traffic on the LSPs available. MATE goal is to find:

$$\min_x C(x) = \min_x \sum_l C_l(x^l) \tag{1}$$

subject to

$$\begin{cases} \sum_{p \in P_s} x_{sp} = r_s & \text{for all } s \in S. \\ x_{sp} \geq 0 & \text{for all } \; p \in P_s \; \text{ and } \; s \in S. \end{cases} \tag{2}$$

A vector x is called **optimal** if it is a minimizer to the problem (1 - 2).

A standard technique to solve the constrained optimization problem (1 - 2) is the gradient projection algorithm. Each iteration in each pair s takes the form:

$$x_s(t+1) = [x_s(t) - \gamma \nabla C_s(t)]^+ \tag{3}$$

where $\nabla C_s(t) = (\partial C(x)/\partial x_{sp}\ ,\ p \in P_s)$ and $[z]^+$ is the projection of a vector z onto the feasible space.

In [6] it is proved that under certain conditions, starting from any initial vector $x(0)$, there exists a sufficiently small stepsize γ such that any accumulation point of the sequence $\{x(t)\}$ generated by the asynchronous algorithm is optimal.

III. MATE algorithm implementation

A. Link cost selection

The link cost used in this work is the one shown as an example in [6]; that is, its delay, modelling it as the average delay of an $M/M/1$ queue. In that case the cost of a link l is

$$C_l(x^l) = R_l(x^l) = \frac{1}{c_l - x^l} \tag{4}$$

Then,

$$\frac{\delta C}{\delta x_{sp}}(x^l) = \sum_{l \in p} C_l'(x^l) = \sum_{l \in p} R_l'(x^l) = \sum_{l \in p} R_l^2(x^l)$$

since $R_l'(x^l) = R_l^2(x^l)$. Hence, it is not necessary to measure x^l in order to calculate $R'(x^l)$, we just need to measure the delay in the link R_l and then calculate R_l^2.

IV. Improvements to MATE for time-varying traffic

MATE is developed under the hypothesis that the average incoming traffic rate to each pair s is constant and known. Generally this is untrue. This fact changes the control problem from a problem of convergence to a fixed optimum point, to a variable optimum point tracking. For that reason, some improvements have to be done to the original algorithm so as it can work independently of the incoming traffic characteristics.

Basically, we introduce two major changes that will be shown in the next sections. But before introducing these changes there is a first detail to modify. That is, as we are working with variable incoming rates, we can no longer work with rates but with percentages. Let $r_s(t)$ be the incoming traffic of a certain pair s, and $\psi_{sp}(t)$ the percentage of the incoming traffic routed on the LSP p, then the update equation using percentages becomes:

$$\psi_{sp}(t+1) = \psi_{sp}(t) - \frac{\gamma}{r_s(t+1)} \sum_{l \in p} R_l^2(x^l). \tag{5}$$

A. Adaptive γ

A.1 Divergence problem when using a fixed γ

In [6] it is proved that, provided that the derivative of the link cost function is Lipschitz with Lipschitz constant L, the algorithm converges to an optimal or a set of optimal points as long as:

$$\gamma < \gamma_{max} = \frac{\phi}{L} \tag{6}$$

where ϕ depends on the network topology and asynchronism degree.

Now, using 4 as our link cost function, the derivative of this function **is not Lipschitz** in the interval $[0, c_l)$, unlike what it is stated in [6], so if we cannot guarantee that the average incoming traffic rate to each link (x^l) is strictly less than its capacity (c_l), we cannot guarantee MATE's convergence. Moreover, although it seems reasonable to make that assumption, it is not sufficient; it can be shown that for any fixed value of $\gamma = \gamma_F$, we can find a value of $x^l_{max} < c_l$ such that $\gamma_{max}(x^l_{max}) < \gamma_F$; thus, we cannot guarantee MATE's convergence, even under the assumption that $x^l < c_l$ for every link l.

Therefore, each pair should use an adaptive value of γ that somehow takes into account in each update the traffic rates in the links belonging to its LSPs. In particular, if in any link l_c (critic link) $x^{l_c} \to c_l$, then γ should tend to zero.

If we analyze the problem intuitively, when the incoming traffic rate of a certain link l is almost equal in media to its capacity, the delays increase so much that the term

$$S_p(\gamma, \vec{R}_p) = \frac{\gamma}{r_s} \sum_{l \in p} \sum_{p \in P_s} R_l^2(x^l) \tag{7}$$

(where $\vec{R}_p$ is the vector of delays in path p) of eq.(5) can be of the order or even greater than the term ψ_p. This leads to the possibility of abrupt changes in the percentages routed to each of the possible paths, which leads to oscillations. For that reason, γ should depend on the incoming traffic rate and the delays measured, which are the only known variables to the algorithm.

We propose the use of an adaptive value of γ calculated in each update, so that the average of the terms $S_p(\gamma, \vec{R}_p)$ is always less than or equal to a given factor ρ of the average of the terms ψ_p. Hence, the variations in the percentages are always less than ρ, avoiding this way abrupt oscillations.

To achieve this goal, we propose the following adaptive calculation of γ:
Defining $Z = \sum_{l \in p} \sum_{p \in P_s} R_l^2(x^l)$ then

$$\gamma = \rho \frac{r_s}{f(Z)}. \tag{8}$$

where $f(Z)$ satisfies that $f(Z) > Z\ \forall Z$, $f(Z) \simeq Z$ for large Z, and $\frac{Z}{f(Z)} \to 0$ when $Z \to 0$. In particular we

can use:

$$f(Z) = Z + \frac{1}{m(1+mZ)} \tag{9}$$

which satisfies all the requirements. Let us explain eq.(8) in more detail. For large delays, Z becomes large, and $f(Z) \simeq Z$, so eq.(8)becomes $\gamma = \rho\frac{r_s}{Z}$. Using that value of γ it is easy to prove that the average of the terms $S_p(\gamma, \vec{R}_p)$ is equal to ρ/N (where N is the number of possible paths), which is ρ times the average of the percentages ψ_p, meeting our goal. For small delays, $Z \to 0$, so $\frac{Z}{f(Z)} \to 0$. It is easy to see that in that case $S_p(\gamma, \vec{R}_p)$ tend to zero, which is exactly what we want, taking into account that when the delays are small we are near the optimum, so the variations should be small.

The parameter m is an adjustment variable which is calculated from the minimal expected delay in a link. Because we are going to have a minimal delay in a link, the value of Z is going to be greater than a Z_{min} which is the value of Z when all the queues of all the routers are empty. If, for example, we want that $f(Z_{min}) = 10Z_{min}$ so that near the optimal point the variations in the percentages of the update are near $\frac{\rho}{10}$, then m becomes: $m \approx \frac{0.2}{Z_{min}}$.

The value of Z_{min} is a configurable parameter of the algorithm, and it can be calculated as $Z_{min} = LR^2_{min}$ where L is the number of links in all the paths, and R_{min} is the minimum delay of a link.

Now, remembering eq.(5), the update equation becomes:

$$\psi_p(t) = \psi_p(t-1) - \rho\frac{\sum_{l\in p} R_l^2(x^l)}{f(\sum_{l\in p}\sum_{p\in P_s} R_l^2(x^l))} \quad \forall p \in P_s \tag{10}$$

This equation has the great advantage that we do not need to know the value of r_s, so it is not necessary to measure the rate of incoming packets.

A.2 Behavior of γ as a function of the traffic.

Now we are going to study the behavior of γ when the traffic in a link l_c is near its capacity.
When $x^{l_c} \to c_{l_c}$, $R_{l_c}(x^{l_c}) \to +\infty$, so $Z \simeq R^2_{l_c}(x^{l_c}) = R_c^2$, and $f(Z) \simeq Z \simeq R_c^2$ obtaining that

$$\gamma = \rho\frac{r_s}{f(Z)} \simeq \rho\frac{r_s}{R_c^2} \to 0$$

meeting our objective. In the same way, γ tends to 0 when we have N critical links.

B. Adaptive interval between updates

When the traffic is not constant the time interval between updates must be adapted to the characteristics of the incoming traffic, which should be quasi-stationary between the updates, in order to stay under the hypothesis of the MATE algorithm.

We propose a procedure that accomplishes that requirement,trying at the same time to minimize the rate of measures, in order to affect as less as possible the network with delay measuring packets.

Suppose we take k measures of the delays $Y_1, \ldots, Y_k$ during the interval dt. These measures are going to be modelled as random variables with the same distribution F and $E(Y_i) = \mu$.

We would like to use a value of k such that:

$$P\Big(\Big|\frac{S_k}{k} - \mu\Big| > \xi\Big) < \epsilon \tag{11}$$

where $S_k := Y_1 + \ldots + Y_k$, and ϵ and ξ are appropriately chosen values.

To obtain k we use the **Chernof Theorem** [7]. Then, in order to satisfy eq.(11), k must satisfy:

$$k \geq \frac{-\ln(\epsilon)}{G(\xi)} \tag{12}$$

where G is the Fenchel-Legendre or the Cramér Transform [7] applied to the random variable $X = Y - \mu$.

Thus, the probability that the average of the k measures fall outside the interval $I_{\mu\xi} := [\mu - \xi, \mu + \xi]$, is smaller than ϵ. On the other hand, we take the reliable interval $I_{\mu\nu} = (\mu - \nu, \mu + \nu)$ where $\nu > \xi$.

The value of μ is estimated in the i-th interval by:

$$\mu_i = \alpha\frac{S_k}{k} + (1-\alpha)\mu_{i-1} \tag{13}$$

where $\alpha \in [0,1]$ and it is configurable.

Our criterion in the selection of the interval between updates is the following:
We take a finite set of possible intervals of duration $dt_1 < dt_2 < \ldots < dt_N$.

Suppose we are at the beginning of the $i+1$-th interval between updates and we have calculated μ_i and the duration of this interval dt_m. We calculate k so that $k \geq -\ln(\epsilon)/G(\xi)$. Then we take k measures of the delays $Y_1, Y_2, \ldots, Y_k$ and calculate their average S_k/k.

Our criterion of adjustment differentiates between four different cases.

- In case $\frac{S_k}{k} > \mu_i + \nu$ we diminish the interval to $dt_{\max\{1,m-1\}}$.
- In case $\frac{S_k}{k} < \mu_i - \nu$, although we can affirm that the statistic changed, it did it for the benefit of the algorithm. So we maintain the same interval dt_m.
- In case $\left|\frac{S_k}{k} - \mu_i\right| \leq \xi$ we are in the stationary situation and so we use the immediately greater interval $dt_{\min\{N,m+1\}}$.
- In case $\xi < \left|\frac{S_k}{k} - \mu_i\right| \leq \nu$, we are in the reliable interval and so we maintain the interval dt_m.

B.1 How do we calculate G without knowing the distribution function F?

To estimate G, we define $X_i = Y_i - \mu$ for all $i \in [1, k]$. Now, we compute

$$\psi(t) = \ln\left[E(e^{tZ})\right] = \ln\left[\frac{1}{k}\left(e^{tX_1} + \ldots + e^{tX_k}\right)\right].$$

$$G(\xi) = \sup\{\xi t - \psi(t) \ : \ t \in J\} \qquad (14)$$

It can be shown that if $X_{min} \leq \xi \leq X_{max}$ then $G(\xi)$ is finite.

If the hypothesis of the Chernof Theorem holds, which means that the random variables $\{Y_i\}_{i=1}^{k}$ (measurements of the delay in the links) are independent and identically distributed with **known** distribution function, this criterion of varying the time between updates works properly. Nevertheless, it is necessary to discuss the veracity of this hypothesis. It is accurate to assume that the random variables are independent because the traffic in a link comes from many independent sources and the time interval between measurements is long enough.

The second hypothesis, that the Y_i are identically distributed, will hold if the time interval between updates is short enough for the system to be quasi-stationary. However, it is possible that the time interval between updates becomes longer than it should for the delay in the link to be quasi-stationary in which case the hypothesis of the Y_j being identically distributed does not hold. It is clear that the longer the interval the more possible that this problem arises.

To solve this problem, we change the computation of the amount of measures to be made as follows:
If the length of the interval is dt_r, then k_r measurements will be made, where:

$$k_r = k_0\left[1 + \frac{1}{2}\left(\frac{dt_r}{dt_1} - 1\right)\right] = \frac{k_0}{2}\left[\frac{dt_r}{dt_1} + 1\right]$$

Then we take all the possible sets of k_0 consecutive measurements and if the mean delay of any of these sets is bigger than $\mu_i + \nu$, the time interval will be shortened. Moreover, the verification of the k_0 consecutive measurements is done in the moment that they are obtained. If we fall outside the reliable interval, the percentages for the load sharing are updated immediately and the next time interval is shortened.

This solves the problem mentioned before diminishing the frequency of measurements when the time interval is lengthened, meeting our objective.

V. Experimental methodology.

An experiment was designed to evaluate the performance of MATE-TV. The objective is to verify that the algorithm converges given an important change in the network incoming traffic, to analyze the evolution of the quantity of measures and the interval length between updates, and to observe the convergence speed. The experiment consisted of two stages and was performed in the network topology of figure 1.

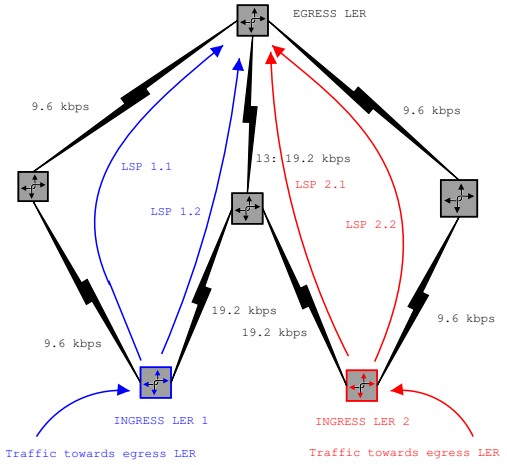

Fig. 1. Experimental topology

A. Stage 1

In this stage there is an incoming traffic in ingress LER 1 and none in ingress LER 2. The incoming traffic has a Poisson distribution with mean 11.5 kbps.The starting percentages assigned to the LSPs of LER 1 are: (LSP 1.1 : 33% ; LSP 1.2 : 67%).

Given that the link capacities of the path LSP 1.2 are twice the capacities of LSP 1.1 the final percentages should be: (LSP 1.1 : 10% ; LSP 1.2 : 90%).

This stage has the purpose of enlarging the interval between updates; despite the fact that there is a slight change in the percentages the delay variation is so small that the interval should increase.

B. Stage 2

In the second stage an incoming traffic to the LER 2 is added with Poisson distribution with mean 23.4 kbps. This overloads the link shared by the paths LSP 1.2 and LSP 2.1, so the MATE-TV running on LER 1 should reduce the interval between updates and start routing more traffic on the LSP 1.1. This stage measures the reaction time of MATE-TV facing a change in the network traffic and the correct convergence to the new optimum point. Once the optimum is reached the interval between updates should increase again.

VI. Results

The following graphics show the results of this experiment. It can be observed that in stage 1 the percentages tended to the predicted values without reaching them since the delay were small and so were the update steps. On the other hand, in this stage the time between updates increased up to 180 seconds.

In stage 2, when the traffic started entering in LER 2 the delays in LSP 1.2 increased abruptly, so the interval between updates was accordingly reduced and the percentages changed to route more traffic on LSP 1.2, reducing the delays. Once reached the optimum point, the interval size started to increase again.

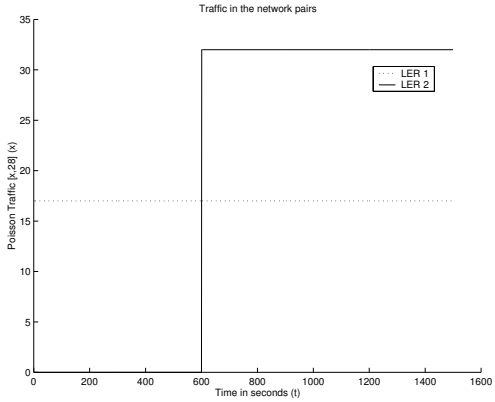

Fig. 2. Incoming traffic by ingress LERs

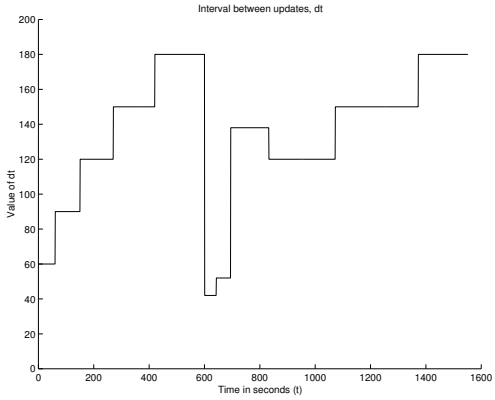

Fig. 3. Evolution of the interval between updates

VII. Conclusions

We have introduced two alterations to the original MATE algorithm in order to remove the assumption of constant average incoming traffic.

- The first improvement is the use of an update step-size which adapts to the traffic characteristics. This allows to guarantee the convergence under time-varying traffic and also improves the convergence speed when the traffic is constant, as explained in IV-A.
- The second improvement is to automatically calculate how many measures to take between the different updates to obtain a reliable estimation of the delays. Besides, the interval between updates is updated, basically taking long intervals in quasi-stationary conditions and shorter intervals when the traffic is changing.

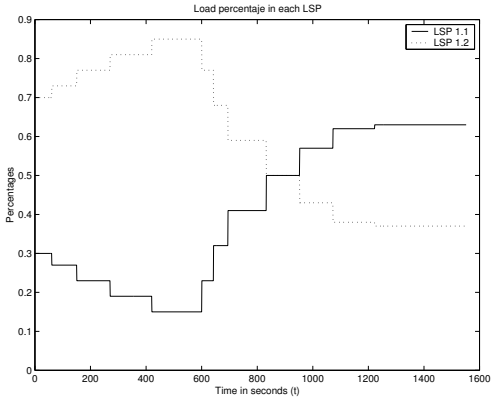

Fig. 4. Percentages forwarded by LSP 1.1 and LSP 1.2

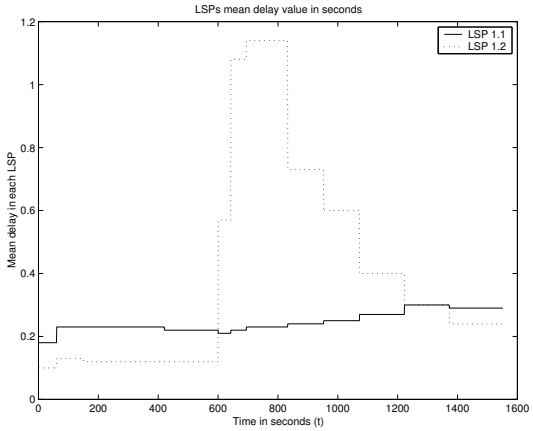

Fig. 5. Mean RTT by LSP 1.1 and LSP 1.2

References

[1] E. Rosen and A. Viswanathan, *Multiprotocol Label Switching Architecture, RFC 3031,* IETF, January 2001

[2] D. Awduche and J. Malcolm *Requirements for Traffic Engineering Over MPLS, RFC2702,* IETF, 1999

[3] R.Casellas and J.L.Rougier and D.Kofman *Packet Based Load Sharing Schemes in MPLS Networks,* ECUMN'2002, Colmar, April 2002

[4] http://sourceforge.net/projects/mpls-linux/

[5] Amir Dembo and Ofer Zeitouni, *Large Deviations Techniques and Applications* Jones and Barlett Publishers, 1993.

[6] IEEE INFOCOM 2001 pag. 1300, *MATE: MPLS Adaptative Traffic Engineering*

[7] James A. Bucklew, *Large Deviation Techniques in Decision, Simulation, and Estimation.*