QoS Routing in MPLS Networks Using Active Measurements

Rahul Mishra, Vinod Sharma
Department of Electrical Communication Engineering
Indian Institute of Science,
Bangalore, 560012 INDIA
080-293-2854
rahulm@pal.ece.iisc.ernet.in, vinod@ece.iisc.ernet.in

Abstract— We consider the problem of selecting an appropriate LSP for a new connection request from several possible LSPs which can satisfy the QoS of the new connection. We propose an active measurement approach where a probing stream is sent along each LSP. Based on the end-to-end delays of the packets of the probing stream, we estimate the mean delays and the throughput the new connection will get. Our approach allows to include the effect of the new connection also in estimating the mean delay and the throughput. The new connection may be using TCP or UDP.

Keywords: **MPLS networks, QoS routing, active measurements.**

1. INTRODUCTION

Multi-Protocol Label Switching (MPLS) architecture [2] was initially motivated by the fact that the layer 2 switching was much faster than layer 3 routing. In MPLS, as in ATM networks, route of a flow is fixed at the time of the connection setup. This route is chosen from one of several possible Label Switched Paths (LSP), based on the Forward Equivalence Class (FEC) of the flow. The FEC could be decided based on the Source-Destination addresses and the QoS (Quality of Service) requirements of the flow. The MPLS network reserves certain resources (buffer space and bandwidth) for any specific FEC on an LSP designated for that FEC. Thus in creating this architecture, many extra features have been added in the IP networks: prespecified path for a flow which remains fixed unless there is a link/node failure on the path, possibility of reserving resources and specifying a path for a flow depending upon its QoS requirements. These turn out to be very useful in providing QoS to different flows. Thus even if, the initial motivation of fast layer 2 switching is no longer there (with the arrival of very fast routers), MPLS architecture is very attractive because of these extra features. Considerable recent effort is being spent (see [1], [3], [5], [6], [7] and reference therein) on the Traffic Engineering aspects of MPLS networks where one tries to satisfy the QoS for different connections by efficient reservation and utilization of resources along different LSPs.

In this paper we consider the following scenario. Depending upon the long term traffic and QoS requirements of flows from different source-destination pairs in an MPLS domain (if MPLS is not end-to-end, one could be considering pairs of ingress and egress LSRs (Label Switched Routers)), various FECs and LSPs have been formed (say using CR-LDP [4]). A flow from an FEC could be sent via several different designated LSPs. Resources (buffers and BW) along the different LSRs on an LSP are reserved for that LSP.

When a new connection request (using TCP or UDP Protocol) arrives in an FEC, then depending upon its QoS requirements and its traffic characteristics, it may require certain BW and buffers on its route. Since the given FEC can use several different LSPs (may be one of them is primary and the other secondary), some of them may be more suitable for the arriving request than others. This is because, at the time the new connection request comes, the different LSPs may be carrying different amount of traffic, i.e., some of them may have more BW available than others. This in general will be time varying and random. A subset (may be not all) of these LSPs may satisfy the QoS requirements of the new connection. One may need to find that subset. If there is more than one LSP in that subset, then the network may have to decide which one of these LSPs to pick. Several studies ([5], [6]) have addressed this problem because it has implications on efficient utilization of network resources and the capability of the network to provide the QoS to different flows. To address these problems it is important to know how much resources are available along an LSP when a new connection request arrives.

We propose a *probing stream* method to estimate the traffic along each LSP. From each of the ingress LSR a low traffic probing stream is sent along each of the LSPs to the egress LSR. Based on the *one way end-to-end delays* on an LSP, we will be able to find the LSPs that can satisfy the QoS requirements of the new request. From this subset of LSPs, using the considerations in [5] and [6], one can pick the 'best' LSP.

The probing stream method is used in many measurement techniques but in the present context of estimating the traffic conditions along the LSPs it has been used in [3] and [7]. It has been pointed out in [3] that it is better to estimate end-to-end delays than to estimate the available BW as suggested in [6]. We will use the approach proposed in [9]. It is substantially different from [3] and [7] and offers extra advantages. For one thing, the probing stream in [9] uses TCP protocol (UDP could also be used but TCP was found to provide better estimates) while in [3] and [7] it uses UDP. In addition, the method in [9] enables one to estimate the throughput and mean delays that the new connection will experience if it uses that LSP, i.e., the effect of traffic of the new connection on itself and other existing flows gets estimated. This is useful because it has been found that in the Internet [10] often a single flow contributes significantly to congestion on a path. The effect of the new connection on the throughput and delays of itself and other connections is not available in other studies. Furthermore, the method in [9] can provide the throughput of a new TCP connection for different TCP parameters (max

window size, propagation delays). It also provides the contribution to the congestion in a bottleneck queue by the overall TCP and UDP traffic separately. It is useful because TCP and UDP traffic behave differently in case of congestion.

The method available in [9] is not suitable for high speed networks. For one thing, it requires one to send two probing streams one after another to estimate two parameters of the overall traffic along a link. To obtain reasonably good estimates, this will cause considerable amount of delay in setting up the new connection. Also, this method will have problems in tracking the time varying traffic statistics that one experiences in an actual network. Thus, in this paper, we modify the scheme in [9] such that only one probing stream is used. It does lead to some degradation in the estimates of one of the traffic parameters mentioned above but does not affect the accuracy of the QoS parameters we eventually need. Furthermore, to enable tracking of time varying traffic statistics, we modify the estimators. We will show that after these modifications the above mentioned drawbacks of the scheme in [9] disappear while retaining its advantages. We will demonstrate the effectiveness of our method via simulations.

Our approach can be used in IP networks in non-MPLS domains also to find routes which provide the needed QoS to a new connection request. This was the driving application for [9]. Our modifications to [9] will be useful in this setup also.

The paper is organized as follows. Section 2 summarizes the approach in [9]. Section 3 provides the changes made in [9] to make it online and also to enable it to track the time varying traffic statistics. Section 4 provides the simulation results. Section 5 gives concluding remarks.

2. Estimating Traffic Parameters

Consider an MPLS domain. It has a set of ingress LSRs and a set of egress LSRs. If a new connection (UDP or TCP) request comes at an ingress node, one needs to find an appropriate LSP from that node to the required egress node. If there is no LSP existing between the ingress and egress node then a new LSP may be established or the packets may be routed at layer 3. If one or more LSPs exist then one needs to find an LSP which provides the required QoS to the new connection request. We will consider minimum throughput and mean delays as QoS requirements. For a TCP connection a minimum throughput guarantee can provide the QoS. For real time connections, a minimum throughput and an upper bound on mean delay can provide some minimal QoS, as a first step. In the following we explain the basic model and the approach in [9].

Let the LSP under consideration has a single bottleneck link. Initially we assume the buffer length at the link to be infinite. We assume that the queue is being shared by various TCP and UDP connections. To find the capability of the LSP to support the QoS requirements of the new connection request, we send a probing stream along the LSP. Figure1 shows the system under consideration. In [9], it is observed that, using TCP for the probing stream leads to better estimation of traffic parameters than using UDP. Therefore we will use TCP for our probing stream.

Let there be N TCP connections passing through the bottleneck link. Let $W_{max}(i)$ be the maximum window size,

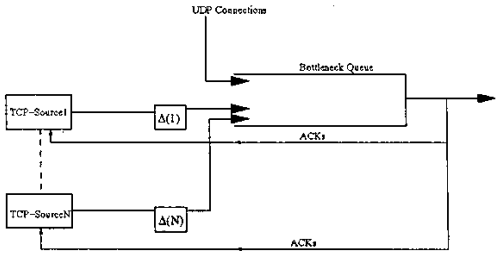

Figure 1. The Model

$\Delta(i)$, the propagation delay and $s(i)$, a generic service time of the ith TCP connection. We assume the packet lengths of a connection to be *iid*. Each UDP connection is supposed to generate a packet stream that can be modeled as an MMPP (Markov Modulated Poisson Process). Since a superposition of MMPPs is also an MMPP, the overall traffic generated by UDPs is also an MMPP. Thus in the following we will consider a single MMPP stream. Packet lengths of the UDP stream are assumed to be *iid* with a general distribution and with generic length s. Mean arrival rate of the MMPP stream is assumed to be $E[\lambda]$. In [8], it is shown that the above system, has a unique stationary distribution for $E[\lambda]E[s] < 1$. The following relations are obtained in [8].

Let for the ith TCP connection, $\lambda_T(i)$ be the throughput in packets/sec., $E[V_T(i)]$ be the stationary mean sojourn time, and $E[q_T(i)]$ be the mean queue length (mean number of its packets in the bottleneck queue). It is shown in [8] that

$$E[V_T(i)] = \frac{\sum_{j=1}^{N} E[s(j)]E[q_T(j)]}{1 - E[\lambda]E[s]}. \quad (1)$$

Also, using Little's law we obtain

$$E[V_T(i)]\lambda_T(i) = E[q_T(i)], \quad (2)$$

$$\lambda_T(i)\Delta(i) = W_{max}(i) - E[q_T(i)]. \quad (3)$$

If $E[D(i)]$ is the mean Round Trip Time (RTT) of the ith TCP connection then we get (including the delays at the nonbottleneck queues in $\Delta(i)$)

$$E[V_T(i)] = E[D(i)] - \Delta(i). \quad (4)$$

If in addition to the above streams, we send a probing stream (TCP) from the ingress node then (1) becomes

$$E[V_T(p)] = (\alpha + E[s(p)]E[q_T(p)])\beta \quad (5)$$

where,

$$\alpha = \sum_{j=1}^{N} E[s(j)]E[q_T(j)], \qquad \beta = \frac{1}{1 - E[\lambda]E[s]} \quad (6)$$

and $E[V_T(p)]$ is the stationary mean sojourn time of the probing stream, $E[s(p)]$ is the mean service time of the probing packets and $E[q_T(p)]$ is the mean queue length of the probing stream at the bottleneck router. By Little's law $E[q_T(p)] = W_{max} - \lambda_T(p)\Delta(p)$. Thus (5) can be rewritten as

$$E[V_T(p)] = (\alpha + E[s(p)](W_{max} - \lambda_T(p)\Delta(p)))\beta. \quad (7)$$

In (6), α indicates the total workload at the queue due to all the existing TCPs (excluding the probing stream) in the bottleneck queue. Also β, gives information about UDP traffic as $(\beta-1)/\beta = E[\lambda]E[s]$ is the total traffic intensity due to UDP connections in the cross traffic at the bottleneck queue.

We send a probing stream and measure the RTTs of the different packets. The sample mean of these RTTs provides an estimate of $E[D(p)]$. The minimum of these RTTs provides an estimate of $\Delta(p)$. Thus we obtain an estimate of $E[V_T(p)] = E[D(p)] - \Delta(p)$. We also obtain an estimate of $\lambda_T(p)$ by dividing the number of probing stream packets sent in time t by t. Plugging these equations in (5), we obtain an equation in α and β. Since we have two unknowns, we need two equations to etimate these variables. By sending two probing streams with different parameters (packet length or W_{max}), one after the other (after all packets of the first stream have been sent and received, the transmission of packets of the second probing stream starts), we obtain estimates of their $E[V_T(p)]$ and $\lambda_T(p)$. Putting these terms in (5) provides an estimate of α and β. Estimated α and β are used in [9] to obtain the throughput and the mean delay (using (1)-(5)) the new connection will receive if it uses the route with the bottleneck router. We will modify this scheme for the present scenario in section 3.

In the above scheme it was required to send two probing streams with different parameters one after another to estimate α and β. Whenever a new connection arrives then probing streams are sent on any potential route to estimate α and β on that route. From the estimated values the throughput and mean delay of the new connection is computed. This leads to problems in tracking of the parameters α and β as the traffic intensity changes. Also the time required to estimate the parameters is large (because of two probing streams). In our present setup we would like to reduce the connection setup time of a new connection. This will be feasible, if we are continuously sending a low traffic probing stream on an LSP and are estimating (tracking) the current values of α and β on that LSP. Thus whenever a new connection request comes, we have the traffic conditions at that time available immediately. To do this we convert the equation (5) into an optimization problem. Using steepest descent algorithm, we will convert the two probing stream approach to a single probing stream approach. This will lead to faster prediction and comparatively reduced amount of bandwidth required due to measurements. We will also be able to track the traffic changes.

3. Extensions And Modifications

To obtain an estimate of α and β from the same probing stream, we try to find the α and β which best fits (7). For this we consider the function

$$g(\mathbf{x}, E[V_T(p)]) = f(\mathbf{x}, E[V_T(p)])^2, \tag{8}$$

where,

$$f(\mathbf{x}, E[V_T(p)]) = (E[V_T(p)] - (\alpha + E[s(p)]E[q_T(p)])\beta), \tag{9}$$

$\mathbf{x} = [\alpha, \beta]^T$ and $E[V_T(p)]$ is the stationary mean sojourn time of the probing stream. The other variables are as in the previous section. We minimise $g(\mathbf{x})$, subject to

$$\alpha \geq 0, \quad \beta \geq 1. \tag{10}$$

These constraints follow from $1 > E[s]E[\lambda] \geq 0$ and $E[s]E[q_T] \geq 0$. Now applying the steepest descent algorithm to the g(x) gives us the iterative algorithm

$$\mathbf{x_{n+1}} = \mathbf{x_n} - \gamma \nabla g(\mathbf{x}), \tag{11}$$

where γ is a small positive constant. This algorithm usually converges to a local minimum. Then we obtain the iterations

$$\alpha_{k+1} = \alpha_k + 2\gamma f(\mathbf{x_k}, E[V_T(p)])\beta_k, \tag{12}$$

$$\beta_{k+1} = \beta_k + 2\gamma f(\mathbf{x_k}, E[V_T(p)]) \\ (\alpha_k + E[s(p)](W_{max} - \lambda_T(p)\Delta(p))). \tag{13}$$

Using simulations we found that taking initial values, $\alpha_0 = 0$ and $\beta_0 = 1$, provide better estimates of α and β.

When network traffic is dynamic in nature, we will need to track the value of α and β in the bottleneck link. For this, we track $E[V_T(p)]$ and $E[\lambda_T(p)]$. We can track mean sojourn time, by updating our estimate frequently with the help of currently measured delays. Denote the mean delay in dynamic traffic environment by $\bar{V}_n$. This is updated at the end of a window W (of the probing TCP) as

$$\bar{V}_{n+1} = (1-\theta)\bar{V}_n + \theta V_n(W) \tag{14}$$

where θ is a fixed small constant and $V_n(W)$ is the average delay experienced by the probing packets during the next window $W(p)$ in the bottleneck queue. We can fix θ, according to the rate of change of traffic i.e. fast changing traffic should have less weight on $\bar{V}_n$ than on $V_n(W)$.

Now replacing $E[V_T(p)]$ in (9) by $\bar{V}_n$ would provide us with the corresponding α and β values at that instant from the limits in (12) and (14).

Prediction of Delay and Throughput

Using α and β estimated from the algorithm in the above section, we can find various QoS parameters obtained by a new connection request on being routed along the given LSP. Let the new connection request be using TCP with the maximum window size W^c_{max} and propagation delay $\Delta(p)$. Let $E[q^c_T]$, $E[s^c]$, $E[V^c_T]$ and λ^c_T be the quantities associated with the new TCP connection. Using (1), (2) and (3) for the new connection, we get

$$E[V^c_T] = (\alpha' + E[s^c]E[q^c_T])\beta, \tag{15}$$

$$E[V^c_T]\lambda^c_T = E[q^c_T], \tag{16}$$

$$\lambda^c_T \Delta^c = W^c_{max} - E[q^c_T], \tag{17}$$

where, $\alpha' = \alpha + E[s(p)](W_{max} - \lambda_T(p)\Delta(p))$. Thus,

$$E[q^c_T] = ((W^c_{max}E[s^c]\beta) - (\alpha'\beta + \Delta^c) + \sqrt{a}) \tag{18}$$

$$/2E[s^c]\beta, \tag{19}$$

$$\lambda^c_T = (W^c_{max} - E[q^c_T])/\Delta^c, \tag{20}$$

$$E[V^c_T] = E[q^c_T]/\lambda^c_T. \tag{21}$$

where,

$$a = ((\alpha'\beta + \Delta^c) - W^c_{max}E[s^c]\beta)^2 + 4E[s^c]W^c_{max}\alpha'\beta^2. \tag{22}$$

Since, α and β are being tracked, we will be able to track the QoS parameters i.e. mean delay and throughput of the new connection along the LSP at any time. Thus we can develop a routing algorithm to find an appropriate LSP for the Best-Effort Traffic.

We can obtain $E[V_U]$, the mean sojourn time of a new UDP connection request (assumed MMPP, with mean arrival rate $E[\lambda']$ and packet lengths *iid*, with a generic service time s') in the respective LSP (with parameters α,β) as

$$E[V_U]' \oplus \beta \alpha + \frac{E[\lambda']E[s']}{1 - E[\lambda']E[s']}, \qquad (23)$$

where, $\alpha' = \alpha + E[s(p)](W_{max} - \lambda_T(p)\Delta(p))$. From (23) one obtains the throughput vs delay relationship of the new UDP connection along an LSP. Hence the $E[\lambda']$ can be changed according to the path or an appropriate LSP can be chosen which could satisfy the QoS of the UDP traffic. This relation although developed for an MMPP stream in [8], actually provides a good approximation for even a CBR UDP stream.

It should be emphasized that in obtaining the QoS parameters for a new connection, the effect of the probing stream and the new connection is included in the above calculations. The effect of the new connection may (if say it is a video stream or a long file) or may not be significant on the QoS of existing connections. In fact recent measurements indicate that often congestion in a link is caused by a single connection. Except in [9] these effects have not been included in the previous studies.

The formulae for QoS of a new TCP connection provided above are when there is no packet loss in the LSP. But if there is, then also we can compute α and β and the other quantities as follows. We have an estimate of $\lambda_T(p)$ and the mean RTT $E[D(p)]$ of the probing stream.If $E[D(p)]\lambda_T(p)$ equals (approximately) $W_{max}(p)$ then there is negligible loss of packets on the LSP and the QoS of the new TCP connection can be obtained as above. In case it is significantly less than $W_{max}(p)$, then there is a nonnegligible loss of packets. Thus, the window size of the new connection will also have $E[W]=E[D(p)]\lambda_T(p)$ (this is because the new connection will also experience the same probability of packet loss and hence the same $E[W]$, irrespective of its W_{max}) and we should use this in place of W^c_{max} in (19)-(22) to compute the QoS parameters.

The above scheme can be used when there is packet loss as in a system with droptail or via a RED (Random Early Detection) control. However, in case of droptail the packets of all TCPs are dropped at the same time (whenever there is buffer overflow) while in RED probabilistically. Thus there is greater correlation in packet drops of different connections in droptail than in RED (in fact this was one of the main motivations to introduce RED). Therefore our predictions of $E[W]$ for the new connection based on that of the probing stream will be more accurate in the droptail case than in the RED control. We observed this in our simulations. Thus, in the next section we have reported only the results of a droptail queuing system.

We canget α and β parameters along each LSP using equations (15)-(22) (for TCP connections) and (23) (for UDP connections). From the estimated α and β at that time, we can obtain the throughput or the mean delay a new (TCP or UDP) connection request will get along each of the LSPs. Then one can pick the LSP which provides the best QoS. In picking the LSP for the new connection, one can also include other considerations ([5], [6]).

4. SIMULATIONS

In this section we report our simulations for the algorithms in Section 3. We have used NS-2.1b8 of UCB/LBNL. The system simulated is depicted in Fig 2. Each simulation is run for 200 seconds. The probing stream uses TCP. We simulated the system when the new connection uses TCP as well as UDP. Because of lack of space we only report the results for TCP. However we obtained good results even for the UDP stream. For TCP we have used the RENO version. The UDP streams were taken as CBR with packet size 500 bytes. The rate of UDP packetsis is also varied which can be obtained from the dotted curve for β shown in Fig 4. We have taken all link capacities except the bottleneck link to be 8Mbps and bottleneck link capacity to be 2.5Mbps. In our simulations γ is 0.1 and θ is 0.8. There were 5 TCP persistent connections. The W_{max} for these connections were 30, 20, 30, 25 and 35 with packet size 500 bytes. The nonpersistent TCP connections arrive as a Poisson process with rate 4 connections/sec. The propagation delay for each TCP connection was 10 msec.. The nonpersistent connections have the file sizes (in packets) and the packet sizes (in bytes), written as an ordered pair (file size, packet size), from the following possible values: (20, 40), (100, 400), (100, 800), (500, 400), (500, 800), (1000, 400), (1000, 800), (1000, 1500). When a new connection arrives it takes any one of these values with probability 1/8.

We first discuss the results for the infinite buffer system.

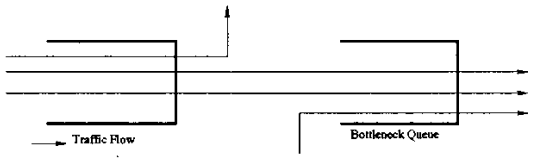

Figure 2. The Simulated Model

The actual and estimated α and β are plotted in Figs 3 and 4 respectively. In all the figures the dotted curves provide the true values while the solid curves provide the estimates.One observes that α is tracked quite well in the whole range but approximation of β is not so good. We found the behaviour of β to be similar for all of our simulations, so we are not providing plots of β for other cases. In [9] the approximation of β was also reasonable. However, as we see from Figs 5 and 6 the prediction of the throughput and mean delay of a new TCP connection are quite accurate and the algorithm is able to track these quite well.

The corresponding results for the finite buffer system with droptail are shown in Figs 7 (α values), 8 (throughput) and 9 (mean delay). The buffer size is 550 packets. Again we observe that except β all other parameters and performance indices are getting tracked quite well.

In the above examples we have kept the number of persistent TCP connections same while nonpersistent TCP connec-

tions arrive and go. This results in a network traffic when after initial transience the traffic parameters and the performance indices tend to settle to some stationary values. In order to show that our algorithm can track well large variations in traffic we present one more example. Here there are no nonpersistent connections. The buffer length is infinite. There are 11 persistent TCP connections and another 23 TCP connections with considerable throughput requirements are opened and closed at different times. The packet lengths, propagation delays of these connections are as above. The W_{max} varies between 20 and 35. These lead to significant variations in α, throughput and mean delays. The corresponding curves are shown in Figs 10, 11 and 12. We observe that even here the predictions match quite well the actual parameters.

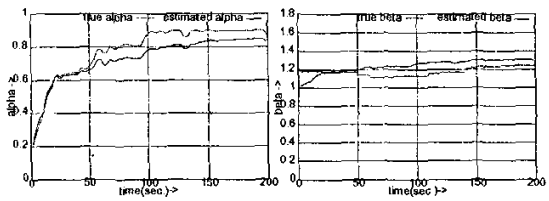

Figure 3. α for Infinite buffers

Figure 4. β for Infinite buffers

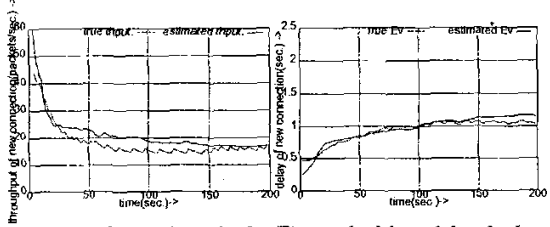

Figure 5. Throughput for Infinite buffers

Figure 6. Mean delay for Infinite buffers

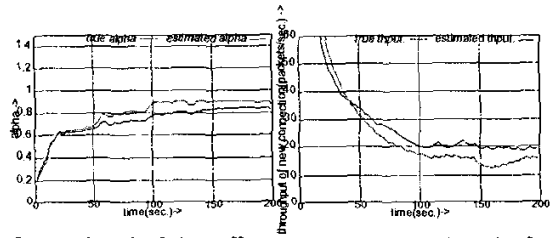

Figure 7. α for finite buffers

Figure 8. Throughput for finite buffers

5. CONCLUSIONS

We have considered the problem of finding an appropriate LSP in an MPLS network which will satisfy the QoS requirements of a new connection request. We use a probing stream along each LSP to keep track of the traffic conditions on the LSPs. Unlike other approaches we can predict the mean delay and throughput a new connection (using UDP or TCP) will receive, taking into account the effect of its own traffic. Using extensive simulations, we have shown that our approach provides reliable estimates.

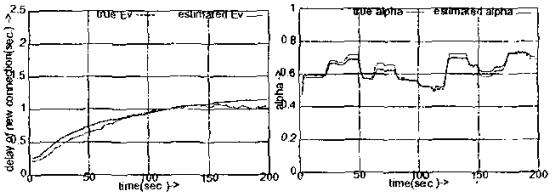

Figure 9. Mean delay for finite buffers

Figure 10. α for Infinite buffers (large variations)

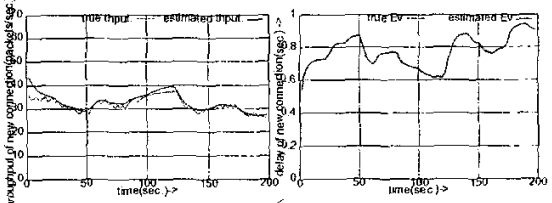

Figure 11. Throughput for Infinite buffers (large variations)

Figure 12. Mean delay for Infinite buffers (large variations)

REFERENCES

[1] T. Anjali, C. Scoglio, J.C. deOliveira, I.F. Akyildiz and G. Uhl, "Optimal policies for label switched path setup in MPLS networks", *Computer Networks* Vol.39,2002,165-183.

[2] U. Black, *MPLS and Label Switching Networks*, Prentice Hall, NJ, 2001.

[3] A. Elwalid, C. Jin, s. Low, I. Widjaja, "MATE: Multipath Adaptive Traffic Engineering", *in Proceedings IEEE INFOCOM 2001.*

[4] B. Jamoussi et al., "constraint-based LSP setup using LDP", *Internet Draft*, sept. 1999.

[5] S. Kamei and T. Kimura, "Evaluation of routing algorithms and network topologies for MPLS traffic engineering" *in Proceedings IEEE GLOBECOM 2001.*

[6] M. Kodialam and T.V. Lakshaman, "Minimum interference routing with applications to MPLS traffic engineering" *in Proceedings IEEE INFOCOM 2000.*

[7] T. Saad, T. Yang, D. Makrakis and V. Groza, "DiffServ-enabled adaptive traffic engineering over MPLS" *in Proceedings of ICII 2001, Beijing.*

[8] V. Sharma and P. Purkayastha,"Performance analysis of TCP connections with RED control and exogenous traffic," *in Proceedings IEEE GLOBECOM 2001.*

[9] V. Sharma and Suma M.B., "Estimating traffic parameters in internet via active measurements for QoS and congestion control" *in Proceedings IEEE GLOBECOM 2001.*

[10] K. Thompson, G.J. Miller and R. Wilder, "Wide-area internet traffic patterns and characteristics", *IEEE Networks*, 6(6), Dec. 1997.