Optimal Traffic Partitioning in MPLS Networks

Esmael Dinan[1], Daniel O. Awduche[2], and Bijan Jabbari[1]

[1] George Mason University,Fairfax, VA 22030, USA
edinan1@gmu.edu, bjabbari@gmu.edu
[2] UUNET (MCI Worldcom), 22001 Loundoun County Parkway,
Ashburn, VA 20147, USA, awduche@uu.net

Abstract. Multiprotocol Label Switching (MPLS) is an emerging Internet technology that facilitates traffic engineering in service provider networks. This paper considers a network performance optimization problem related to traffic engineering over MPLS. We investigate the issue of dynamic partitioning of MPLS ingress traffic into several parallel Label Switched Paths (LSP). Specifically, we present a stochastic framework for the traffic partitioning problem. Within this framework, a set of parallel edge disjoint LSPs is modeled by parallel queues and a partitioning algorithm is devised for different service classes that is adaptive to the prevailing state of the network. The performance of this approach is illustrated by numerical examples.

1 Introduction

As the Internet continues to grow exponentially and becomes more mission critical, the need for performance optimization of service provider networks through effective and efficient resource management becomes imperative. Performance optimization of operational networks is achieved through traffic engineering. The basic problem of traffic engineering in service provider networks concerns the mapping of traffic onto the network infrastructure, such that traffic oriented and resource oriented performance objectives are achieved. Traffic oriented performance characteristics are usually expressed in terms of QoS parameters, such as loss, delay, and goodput. On the other hand, resource oriented performance objectives relate to the efficient utilization of network assets.

Multiprotocol Label Switching (MPLS) is one of the most promising technologies for traffic engineering in IP networks [2]. MPLS is an approach to packet forwarding whereby short fixed length labels are attached to packets by a label switching router (LSR) at the ingress to an MPLS domain [1]. An ingress LSR attaches labels to packets based on the concept of forwarding equivalence classes (FEC), so that packets that belong to the same FEC are assigned the same label value. As labeled packets traverse the MPLS domain, packet forwarding is performed according to the classical label swapping paradigm. MPLS becomes a powerful abstraction for traffic engineering when coupled with a signaling protocol that supports explicit LSP setup capability [3].

G. Pujolle et al. (Eds.): NETWORKING 2000, LNCS 1815, pp. 144–155, 2000.

The requirements for traffic engineering over MPLS are outlined in [2]. These requirements include a set of capabilities that facilitate the realization of a variety of network optimization policies. Regardless of the specific optimization policy employed in a given network, three fundamental problems related to traffic engineering over MPLS were identified in [2]. The first problem concerns how to partition traffic into traffic trunks or forwarding equivalence classes. The second problem concerns how to map traffic trunks onto label switched paths (LSPs). The third problem concerns how to map traffic trunks onto the physical network topology through the selection of routes for LSPs.

This paper describes an analytical approach to network performance optimization which relates to the first two aspects of the MPLS traffic engineering problem. Specifically, we present a stochastic framework for dynamic traffic partitioning by which more efficient use of alternate paths can be made in MPLS domains. We suppose that a set of parallel edge disjoint LSPs have been preestablished between an ingress node and an egress node. We describe a procedure for partitioning ingress traffic into forwarding equivalence classes, and a traffic assignment algorithm for mapping the FECs onto the pre-established parallel LSPs, taking into account the dynamics of network state.

The traffic partitioning problem as presented here is peculiar to MPLS and is distinguished from ATM based traffic management mechanisms. A specific approach to MPLS adaptive traffic engineering (called MATE) based on a partitioning paradigm was presented in [8]. The MATE proposal maps input traffic to different LSPs based on system parameters measured by probe packets.

This paper is organized as follows: Section 2 describes the networking context and associated system assumptions. Section 3 presents the analytical model, algorithms, and performance analysis. Transient behavior of the system is studied in section 4. Section 5 illustrates the concepts through numerical examples. Finally, section 6 provides concluding remarks.

2 The Traffic Partitioning Problem in MPLS Networks

We consider an MPLS network in which the input traffic to an ingress label switching router (LSR) is dynamically partitioned and mapped onto several label switched paths. Figure 1 depicts an example of such a network. Given this network, the problems that we address in this paper are two fold. The first aspect concerns the partitioning of ingress traffic into FECs. The second aspect concerns the mapping of FECs onto pre-established LSPs. The intent is to perform the partitioning and mapping operations in such a way that traffic oriented performance characteristics are enhanced, while network resources are utilized efficiently. Since the characteristics of ingress traffic, the membership of FECs, and state of the network vary with time, the optimal partitioning procedure needs to be dynamic and adaptive.

In a differentiated services Internet, the ingress traffic will consist of a collection of traffic aggregates belonging to different service classes with different forwarding treatment requirements which are specified in terms of per-hop for-

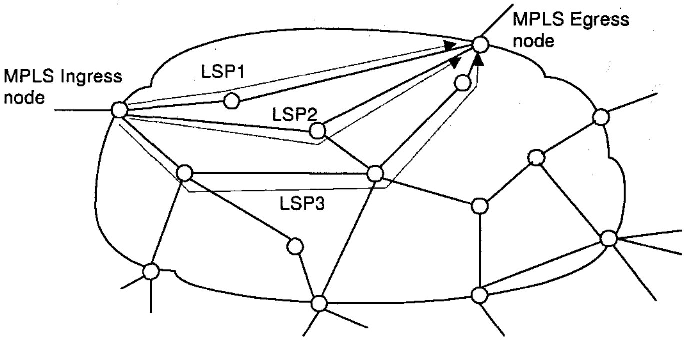

Fig. 1. An example MPLS network.

warding behaviors (PHB) encoded in the packet headers (see e.g., [4]). In the following, the performance measures for transmission quality will be the probability of packet loss and packet transmission delay. The traffic mapping is done in two stages. First, conceptually, the input traffic is partitioned into N bins at the MPLS ingress node, each with approximately $r = \lambda/N$ packets/sec. Then the traffic from the N bins is mapped onto n pre-established LSPs. With this method, the partitioning ratio is a discrete variable, i/N, $1 \leq i \leq N$. A simple method to partition the input traffic is on a per-packet basis, for example in a round-robin fashion. However, this method suffers from the possibility of excessive packet reordering and is not recommended in practice. Another well known method is to filter the ingress using a hash function on a subset of the IP packet header (see e.g., [5]).

3 System Model and Analysis

We focus on a single MPLS ingress LSR which is connected to an egress LSR via n parallel LSPs. Let 's denote such set of LSPs by $\Psi = \{\varphi_i : i = 1, ..., n\}$. We model each hop through which an LSP traverses by a queue (specifically an M/M/1/K queue), so that each LSP is effectively represented by a sequence of such queues. As shown in Figure 2, the system essentially consists of a network of queues. Let $Q = \{Q_{ij}; i = 1, ..., n, j = 1, ..., R_i\}$ be the set of queues comprising the queueing network, where R_i is the number of hops traversed by $\varphi_i \in \Psi$. Let $\lambda(t)$ be the time-dependent aggregate arrival rate at the ingress LSR of the parallel LSPs destined for the egress LSR. We assume that the traffic arrival rate at each nodal queue traversed over LSP $\varphi_i \in \Psi$ is a function of time and can be divided into two parts: (1) the flow rate $\lambda_i(t) = p_i\lambda(t)$, which is the time-dependent contribution of φ_i and takes into account the traffic mapping proabability vector $P = \{p_i : i = 1, ..., n; \sum_{i=1}^{n} p_i = 1\}$; $\lambda_{ij}(t)$ represents this flow passing through Q_{ij} and (2) the variable $\gamma_{ij}(t)$, which represents the time-dependent contribution from all other LSPs that traverse the node. Let μ_{ij} be

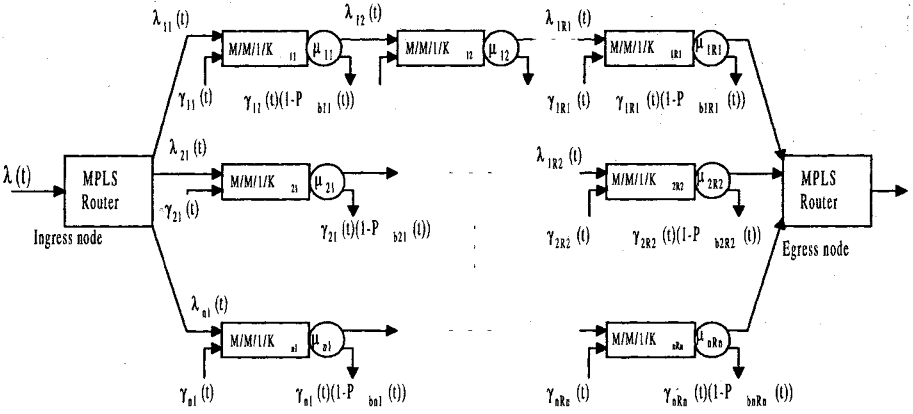

Fig. 2. System model considering a series of queues for each LSP.

the service rate associated with queue j on LSP φ_i. The available bandwidth at each queue is the difference between the average service rate and the average aggregate arrival rate of all traffic that traverse the queue. The methods for measuring the available bandwidth of a given path have been described in [6]. We suppose that the variation of $\lambda(t)$ and $\gamma_{ij}(t)$ $, 1 \leq i \leq n, 1 \leq j \leq R_i$, are slow enough with respect to time, so that we can consider only the steady-state behavior of the queues in the analysis.

3.1 Best Effort Traffic

First, we consider only best effort traffic. We define the vector

$\Lambda(t) = \left[\lambda_1(t), \lambda_2(t), ..., \lambda_n(t)\right]^T$ as the input traffic rates to LSPs $\varphi_i \in \Psi$, and $\Lambda_i(t) = \left[\lambda_{ti1}(t), \lambda_{ti2}(t), ..., \lambda_{tiR_i}(t)\right]^T = \Lambda_i + \Gamma_i$ where $\lambda_{tij}(t) = \lambda_{ij}(t) + \gamma_{ij}(t)$, as the vector of queue arrival rates and $M_i = \left[\mu_{i1}, \mu_{i2}, ..., \mu_{iR_i}\right]^T$ as the set of queue service rates. The system performance for best effort is measured by the total number of lost packets. The objective of the analysis is to find the traffic mapping vector $P(t) = \left[p_1(t), p_2(t), ..., p_n(t)\right]^T$ that optimizes the system performance. We have

$$\left[\lambda_1(t), \lambda_2(t), ..., \lambda_n(t)\right]^T = \left[p_1(t), p_2(t), ..., p_n(t)\right]^T \lambda(t) \tag{1}$$

and

$$\sum_{i=1}^{n} p_i(t) = 1 \tag{2}$$

We need to find the column vector $P(t)$ such that the loss rate

$$L(t) = P_{b1}(t)\lambda_1(t)+P_{b2}(t)\lambda_2(t)+...+P_{bn}(t)\lambda_n(t) = \sum_{i=1}^{n} P_{bi}(t)\lambda_i(t) = P_b(t)^T \Lambda(t) \tag{3}$$

is minimized, where $P_b(t) = [P_{b1}(t), P_{b2}(t), ..., P_{bn}(t)]^T$. The parameter P_{bi} is the average probability of loss for the traffic of LSP φ_i, Λ_i. We define P_{bij} as the probability of loss in queue Q_{ij} with the arrival rate $\lambda_{tij}(t)$. The loss rate of LSP φ_i traffic in this queue is equal to

$$L_{ij}(t) = P_{bij}(\lambda_{tij}(t), \mu_{ij})\lambda_{ij}(t)$$

$$= P_{bij}(\lambda_{tij}(t), \mu_{ij})\lambda_{i(j-1)}(t)\left(1 - P_{bi(j-1)}(\lambda_{ti(j-1)}(t), \mu_{i(j-1)})\right) \tag{4}$$

where $\lambda_{ij}(t) = \lambda_{i(j-1)}(t)\left(1 - P_{bi(j-1)}(\lambda_{ti(j-1)}(t), \mu_{i(j-1)})\right)$ and $\lambda_{i1}(t) = p_1(t)\lambda(t)$.

Assuming blocking to be relatively small in the network ($P_{bij} << 1$), the departure of a queue with Poisson arrival rate can be approximated by another Poisson process. Therefore, each queue in the system can be approximated by an equivalent M/M/1/K queue. The total loss rate, $L(t)$, can be rewritten as follows

$$L(t) = \sum_{i=1}^{n}\left(\sum_{j=1}^{R_i} P_{bij}(\lambda_{tij}(t), \mu_{ij})\lambda_{ij}(t)\right) \tag{5}$$

where for a fixed time t_0, K_{ij}, μ_{ij} and $\gamma_{ij}(t_0), 1 \leq i \leq n$ and $1 \leq j \leq R_i$, are known. The variables $\lambda_i(t), 1 \leq i \leq n$, are to be determined such that $\lambda(t) = \sum_{i=1}^{n} \lambda_i(t)$, and $L(t)$ is minimized. The set $\{\lambda_i(t), 1 \leq i \leq n\}$ that minimizes $L(t)$ can be found by iteration or Lagrange Multiplier methods [7].

Since the queue arrival rate vectors $\Gamma_i(t)$, and the Ingress node arrival rate $\lambda(t)$ are functions of time, an iterative algorithm needs to be executed in every time interval Δt, hence a significant computational effort is involved. Therefore, we present a simple method to calculate the optimal vector $p(t + \Delta t)$ when the value of the vector $p(t)$ is known at time t. With this technique, the minimum value of $L(t)$ needs to be obtained just one time, and then $p(t)$ is updated periodically with a simple algorithm in each time step Δt.

$L_i(\lambda_i(t), M_i)$ represents the loss rate in the path φ_i and is equal to

$$L_i(\lambda_i(t), M_i) = P_{bi}(t)\lambda_i(t) = \sum_{j=1}^{R_i} P_{bij}(\lambda_{tij}(t), \mu_{ij})\lambda_{ij}(t) \tag{6}$$

We define function $S_{\Gamma_i}(t)$ as the sensitivity of the loss rate to the other paths traffic rate, and the function $S_{\lambda_i}(t)$ as the sensitivity of the loss rate to the path arrival rate, where the following holds

$$: S_{\Gamma_i}(t) = \left[\frac{\partial L_i(\lambda_i(t), M_i)}{\partial \gamma_{i1}(t)}, \frac{\partial L_i(\lambda_i(t), M_i)}{\partial \gamma_{i2}(t)}, ..., \frac{\partial L_i(\lambda_i(t), M_i)}{\partial \gamma_{iR_i}(t)}\right] \tag{7}$$

$$S_{\lambda_i}(t) = \frac{\partial L_i(\lambda_i(t), M_i)}{\partial \lambda_i(t)} \tag{8}$$

In a simplified model where each LSP is modeled by just one M/M/1/K queue (see Figure ??), which presents the bottleneck of the path, we have:

$$S_{\lambda_{pi}}(t) = S_{\lambda_i}(t) = \frac{\partial L_i(\lambda_i(t), \mu_i)}{\partial \lambda_i(t)}$$

$$= \rho_i(t)^{K_i} \frac{(1+K_i)\left(1-\rho_i(t)^{K_i+2}\right) - (2+K_i)\left(\rho_i(t) - \rho_i(t)^{K_i+2}\right)}{\left(1-\rho_i(t)^{K_i+1}\right)^2} \tag{9}$$

Since the loss rate increases with increasing arrival rate, the elements of $S_{\Gamma_i}(t)$ and $S_{\lambda_i}(t)$ are positive.

We now consider two different cases. First, we consider the effect of the variation of the vectors $\Gamma_i(t)$ on the mapping vector, $P(t)$. Then, we consider the effect of the variation in the Ingress node arrival traffic rate $\lambda(t)$. In the simplest case, we suppose that during time Δt only the arrival rate of queues in LSP φ_i changes, and we have

$$\Gamma_i(t+\Delta t) = \Gamma_i(t) + \Delta\Gamma_i(t),\ \Delta\Gamma_i(t) = \left[\Delta\gamma_{i1}(t), \Delta\gamma_{i2}(t), ..., \Delta\gamma_{iR_i}(t)\right]$$

$$\Gamma_k(t+\Delta t) = \Gamma_k(t), 1 \leq k \leq n, k \neq i \tag{10}$$

To compensate for the effect of variation of the arrival rates on the $L_i(.,.)$, we need to change the arrival rate of the LSP φ_i in a way that loss rate remains unchanged. This implies that:

$$L_i(\lambda_i(t+\Delta t), M_i) = L_i(\lambda_i(t), M_i) + \sum_{j=1}^{R_i} S_{\Gamma_{ij}}(t)\Delta\gamma_{ij}(t) + S_{\lambda_i}(t)\Delta\lambda_i$$

$$= L_i(\lambda_i(t), M_i) \tag{11}$$

where $\lambda_i(t+\Delta t) = \lambda_i(t) + \Delta\lambda_i$, and linear approximation has been used for the loss rate changes during time interval Δt. Then,

$$\Delta\lambda_i = -\frac{\sum_{j=1}^{R_i} S_{\Gamma ij}(t)\Delta\gamma_{ij}(t)}{S_{\lambda_i}(t)} \tag{12}$$

Therefore, to compensate for the effect of $\Delta\Gamma_i(t)$ of the queues of LSP$_i$, we change the arrival rate of LSP φ_i by $\Delta\lambda_i$ (that may be positive or negative). Now, we need to find a method to distribute the extra traffic $-\Delta\lambda_i$ (let's say *delta traffic*) on different paths. We present the following *Traffic Assignment Algorithm* which can be used to map the traffic $(-\Delta\lambda_i)$ to a path such that $L(t)$ remains minimum:

```
Traffic Assignment Algorithm for the delta traffic ($-\Delta\lambda_i$):
if ($-\Delta\lambda_i$) $> 0$
        Shift the traffic ($-\Delta\lambda_i$) from path $\varphi_i$ to the path with the lowest value of $S_\lambda(t)$.
else (($-\Delta\lambda_i$) $< 0$)
        Shift the traffic ($\Delta\lambda_i$) from the path with the highest value of $S_\lambda(t)$ to path $\varphi_i$.
end
```

The main idea is that if $(-\Delta\lambda_i) > 0$, we add the delta traffic to the path whose loss rate is the least sensitive to λ, and therefore we minimize the increase of P_b. If $(-\Delta\lambda_i) < 0$, we decrease the traffic of the path whose loss rate is the most sensitive to λ, and therefore we maximize the decrease of P_b.

Now, consider a more complex case, wherein not only the arrival rate of all queues but also the arrival rate to the ingress node change during time interval Δt, i.e., $\Gamma_i(t + \Delta t) = \Gamma_i(t) + \Delta\Gamma_i(t)$, where $1 \leq i \leq n$, and $\lambda(t + \Delta t) = \lambda(t) + \Delta\lambda(t)$.

The following algorithm re-assigns the input arrival traffic to different queues:

1. Calculate $\Delta\lambda_i = \frac{\sum_{j=1}^{R_i} S_{\Gamma_{ij}} \Delta\lambda_{ij}}{S_{\lambda_i}(t)}$ for $1 \leq i \leq n$.

2. Assign the $n+1$ delta arrival traffic ($\Delta\lambda$ and $\Delta\lambda_i, 1 \leq i \leq n$) to the queues as follows:

    For each delta arrival traffic:
    a. Execute the *Traffic Assignment Algorithm.*
    b. Update variables $P(t)$ and $\Lambda(t)$.

The above algorithm assigns the traffic to the queues in a way to keep the loss rate minimum. The closeness of the partitioning vector obtained by the above algorithms and the optimum one depends on the error of the linear approximation used in Eq. ??. Smaller changes during time interval Δt lead to more exact results. If the number of traffic bins shifted in each run of the algorithm is higher than a certain value ($\Delta\lambda_{\max}$), the time interval Δt can be decreased in order to obtain a better approximation. The parameter Δt can be controlled dynamically based on the system parameter variation in order to obtain exact results. It is notable that (as described in section II), the elements of the vector $P(t)$ can only have discrete values, i/N, $1 \leq i \leq N$, where N is the number of bins that the input traffic is mapped. Therefore, when $\Delta\lambda$ is calculated, it should be rounded to a discrete number of bins.

3.2 TCP Traffic

Another interesting case is when both delay and loss are important factors in system performance optimization. A good example is TCP traffic, which is an important type of traffic in contemporary data networks. Since TCP reacts to packet losses, the total system goodput and packet delay are affected by both delay and loss. It is well-known that TCP achieves bandwidth that is inversely proportional to the square root of the packet loss probability $P_b(t)$, under idealized conditions. Note that this approximation is reasonable in a certain range of P_b (not too close to zero or one). Since the bandwidth achieved by TCP is reduced by a factor of square root of P_b, the total delay will be proportional to $\sqrt{P_b}\tau$. In this case the optimal mapping vector should be calculated to minimize

$$\tau_{avg} = \sqrt{P_{b1}(t)}\tau_1(t)P_1(t) + ... + \sqrt{P_{bn}(t)}\tau_n(t)P_n(t) = \sum_{i=1}^{n}\sqrt{P_{bi}(t)}\tau_i(t)P_i(t) \tag{13}$$

where

$$\tau_{tcp,i}(\lambda_i(t),\mu_i) = \sqrt{P_{bi}(\lambda_i(t),\mu_i)}\tau_i(\lambda_i(t),\mu_i)P_i(t) \tag{14}$$

And the sensitivity functions will be

$$S_{\Gamma_i}(t) = \left[\frac{\partial\tau_{tcp,i}(\lambda_i(t),M_i)}{\partial\gamma_{i1}(t)}, \frac{\partial\tau_{tcp,i}(\lambda_i(t),M_i)}{\partial\gamma_{i2}(t)}, ..., \frac{\partial\tau_{tcp,i}(\lambda_i(t),M_i)}{\partial\gamma_{iR_i}(t)}\right] \tag{15}$$

$$S_{\lambda_i}(t) = \frac{\partial\tau_{tcp,i}(\lambda_i(t),M_i(t))}{\partial\lambda_i(t)} \tag{16}$$

The algorithm described previously can be used for TCP traffic partitioning with the above parameters.

4 Transient Behavior and System Stability

Now let us consider the transient response of the system when a link is shared by more than one LSP (as an example see Figure 5). Traffic assignment algorithm is executed in all Ingress nodes of the MPLS network. The MPLS Ingress nodes dynamically partition the input traffic based on the current state of the network. Furthermore, there is no need for time synchronization among Ingress nodes in the algorithm. Since a link might be shared by more than one label switched path, a change of traffic partitioning table in one Ingress node can be interpreted as a change of network state by other Ingress nodes. After Δt, these Ingress nodes will execute the traffic assignment algorithm and change again their traffic partitioning table. At a stable equilibrium point any small excursion of the system state variables is forced back to another equilibrium values. Otherwise, for an unstable equilibrium point, a perturbation of the state variables is forced further away from an equilibrium point. The model presented in Figure 2 and the developed analysis let us to follow the transient and steady state behavior of different architectures due to any input traffic or system state changes.

Let N be the number of Ingress nodes in the system, and $\Lambda_I = [\lambda_{I1},..., \lambda_{IN}]$ the input rate to the Ingress nodes. We suppose that during time Δt the arrival rate of Ingress nodes changes by $\Delta\Lambda_{In} = [\Delta\lambda_1,..., \Delta\lambda_2]$. As described in section III, the arrival rate of each nodal queue in LSP i of Ingress node k is a function of time and can be divided into two parts: traffic of LSP i, $(\lambda_{ij})_k$, which is a part of the arrival rate traffic λ_{Ik}, and traffic of other paths input to the node, $(\gamma_{ij})_k$.

Other paths traffic (or load traffic) represents the time dependent traffic load of the node. To calculate the *load traffic* of each link, we suppose that packet loss is negligible. Therefore, we will have the following relation for the load traffic, $\left(\gamma_{ij}\right)_k$, which represents load traffic of link j of LSP$_i$ of Ingress node k.

$$\left(\gamma_{ij}\right)_k = \sum_{k=1}^{N_I} \sum_{n=1}^{LSP_k} a_{kn} \left(\lambda_{In}\right)_k \tag{17}$$

where $a_{kn} = \begin{cases} 1, \text{ If the link is shared by LSP } n \text{ of ingress node } k \\ 0, \text{ If the link is not shared by LSP } n \text{ of Ingress node } k \end{cases}$, LSP$_k$ is the number of paths originated from Ingress node k, and $\left(\lambda_{In}\right)_k$ is the arrival traffic of LSP n of Ingress node k.

If during Δt, $\Delta(\gamma_{ij})_k = \sum\limits_{k=1}^{N_I} \sum\limits_{n=1}^{LSP_k} a_{kn}\Delta\left(\lambda_{In}\right)_k$ or $\Delta\lambda_{Ik}$ has a non-zero value, the ingress node k will execute the Traffic Assignment Algorithm. There is no need for synchronization between different nodes. A small change in network state might be able to move the network to a unstable point. We show here that the algorithm moves the system to a stable point. That means that a change in the network after time Δt is reflected to the same node by always a smaller value.

We consider the case where the traffic belongs to only the best effort traffic . In LSP i of Ingress node k, the packet loss rate is more sensitive to the variation of the LSP traffic $\left(\Delta\lambda_{Ik}\right)_k$ than the load traffic of a link of this node, since the LSP traffic passes through all the nodes of an LSP. We have

$$S_{\lambda_i}(t) = \frac{\partial L_i(\lambda_i(t), M_i)}{\partial\lambda_i(t)} > S_{\Gamma_{ij}}(t) = \frac{\partial L_i(\lambda_i(t), M_i)}{\partial\gamma_{ij}(t)} \tag{18}$$

or $\frac{S_{\Gamma_{ij}}(t)}{S_{\lambda_i}(t)} < 1$ and for the assigned traffic

$$|\Delta\lambda_i| = \left| -\frac{\sum\limits_{j=1}^{R_i} S_{\Gamma_{ij}}(t)\Delta\gamma_{ij}(t)}{S_{\lambda_i}(t)} \right| = \left| -\sum_{j=1}^{R_i} \frac{S_{\Gamma_{ij}}(t)}{S_{\lambda_i}(t)}\Delta\lambda_i(t) \right| < \left| \sum_{j=1}^{R_i} \Delta\gamma_{ij}(t) \right|$$

This shows that the changes in the input traffic of an LSP is less than the sum of all changes in the load traffic. Changes in load traffic represents the changes in traffic partitioning of other Ingress nodes in the network. We mention here that this traffic is directed toward a link with the lowest sensitivity. That also helps to reduce the amount of the traffic variation, and to provide system stability.

5 Numerical Analysis and Discussion

In the following numerical analysis, we restrict attention to three parallel LSPs for the system. Each LSP is modeled by one queue with the service rate of 4 K packets/s. The queue sizes are considered to be equal to 15, 10 and 7, and the arrival rate to the ingress node is 2 K packets/s. We consider a hashing function

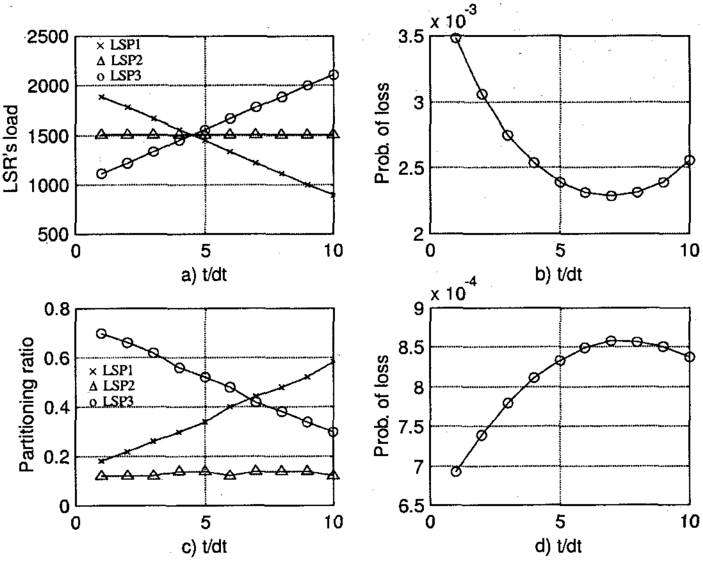

Fig. 3. Traffic partitioning vector for the best effort traffic, a) LSR's load, λ_{li}, b) P_b with equal partitioning, c) optimal partitioning vector, d) P_b with optimal partitioning.

that divides the input traffic into 100 bins. Figure 3.a shows LSR's load (γ_i) and their variations in time. The available capacity of LSP$_1$ increases with time, LSP$_2$ has a constant load, and LSP$_3$ is congested with time.

Figure 3.b shows the system performance measured by average P_b in the system when the input arrival is divided into three equal parts and the partitioning vector is fixed during the time. Figure 3.c shows the optimal partitioning vector in time obtained by the presented algorithm in order to minimize P_b, and Figure 3.d plots the average P_b when the optimal partitioning algorithm has been applied in the system. Plots in Figure 3 show that the partitioning vector not only depends on the system load but also depends on the queue sizes. In a general term, when the other path arrival rate of a node decreases, it can accept more traffic. Also a queue with larger size can support more traffic with the same probability of blocking. For example, consider the case when $\gamma_1 = \gamma_2 = \gamma_3$ in figure 3.a, the vector P at this time is equal to $P = [0.32\ 0.14\ 0.54]$. It can be seen that optimal partitioning improves the system performance by more than 3 times. Figure 4 shows the same parameters when traffic is considered to be transported by TCP. The optimum partitioning vector is different from the one obtained in Figure 3. This shows that traffic assignment parameters depend not only on the network state but also on the QoS requirements. To illustrate the transient behavior of the network, an example is considered in Figure 5. For each link, as we described before, an Ingress node considers the traffic of its own established path as the *assigned traffic*, and the traffic directed by other Ingress

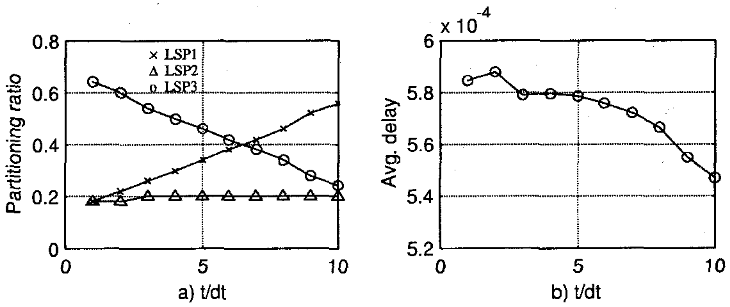

Fig. 4. Traffic partitioning vector for TCP traffic (with the arrival rates as shown in Figure 4.a), a) optimal partitioning vector, b) average delay with optimal partitioning.

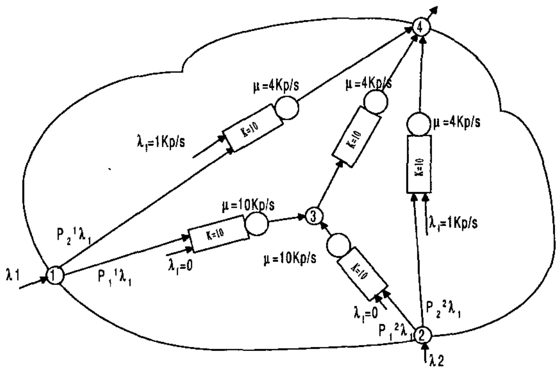

Fig. 5. An example in order to study stability and the transient behavior of an MPLS network.

nodes to this link as the current *load traffic*. Two LSPs, whose traffic is controlled by different Ingress nodes, share the link between nodes 3 and 4. Main system parameters as service rates and arrival rates are shown in the figure. System starts with $\lambda_1 = 3$ K packets/s and $\lambda_2 = 0$ and after Δt, input traffic of node 2 is increased to $\lambda_2 = 1$ K packets/s. The simulation results are given in Figure 6, for the Best Effort Traffic. When the arrival traffic of node 2 increases, it directs some part of the traffic to node 3. As a result, Ingress node 1 sees a change in network state and reduces the traffic directed toward node 3, which can be interpreted as a network state change for node 2. This loop continues until the system reaches to a stable point.

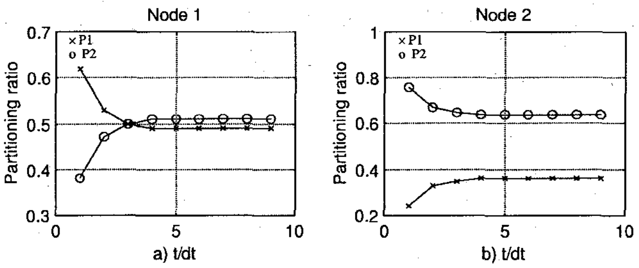

Fig. 6. Traffic partitioning parameters a) Node 1, b) Node 2.

6 Conclusion

This paper described an approach to network performance optimization concerning traffic engineering over MPLS. We developed an analytic model to investigate the optimal partitioning of MPLS ingress traffic into parallel label switched paths, based on the current state of the network. Each LSP was modeled by a series of queues. A stochastic framework and a partitioning algorithm were presented, which take into account the dynamics of the network state. An Algorithm was exhibited for the input traffic assignment. The system performance improvement was illustrated by numerical results. The results suggest that the optimal partitioning of input traffic increases the system performance. The efficacy of this approach depends on QoS requirements.

References

1. E. C. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," work in progress, Internet Draft <draft-ietf-mpls-arch-06.txt>, August 1999.
2. D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus, "Requirements for Traffic Engineering Over MPLS," Internet RFC 2702, Sep. 1999.
3. D. Awduche, L. Berger, D. Gan, T. Li, G. Swallow, and V. Srinivasan, "Extensions to RSVP for LSP Tunnels," work in progress, Internet Draft <draft-ietf-mpls-rsvp–lsp-tunnel-02.txt>, March 1999.
4. V. Jacobson, K. Nichols, K. Poduri, "An Expedited Forwarding PHB," RFC 2598, June 1999.
5. C. Villamizar, "OSPF Optimized Multipath (OSPF-OMP)," work in progress, Internet Draft <draft-ietf-ospf-omp-02.txt>, February 1999.
6. S. Keshav, "A Control-Theoretic Approach to Flow Control," *Proc. SIGCOM'91*, pp. 3-15, Sep. 1991.
7. W. L. Winston, *Operation Research, Application and Algorithms*, Duxbury Press, 1994.
8. I. Widjaja and A. Elwalid, "MATE: Adaptive Traffic Engineering," work in progress, Internet Draft <draft-widjaja-mpls-mate-00.txt>, August 1998.
9. T. Li, G. Swallow, and D. Awduche, "IGP Requirements for Traffic Engineering with MPLS," Internet Draft <draft-li-mpls-igp-te-01.txt>, February, 1999.