Journal of Computer Science 4 (3): 232-238, 2008
ISSN 1549-3636

Enhanced Variable Splitting Ratio Algorithm for Effective Load Balancing in MPLS Networks

[1]G. Murugesan, [2]A.M. Natarajan and [1]C. Venkatesh
[1]Kongu Engineering College, Perundurai, Erode - 638052, Tamilnadu, India
[2]Bannari Amman Institute of Technology, Sathiamangalam, Tamilnadu, India

Abstract: Fast load balancing is of the highest importance to utilize the resources effectively and improve the performance of the Multi Protocol Label Switching (MPLS) enabled Internet Protocol (IP) networks. Load balancing splits a part of traffic called splitting ratio from the maximum loaded path in to minimum loaded path. One very strong constrain in balancing the load is the tradeoff between the delay to reach the balanced state and the oscillations generated by splitting the load at the balanced state. To avoid oscillations, the splitting ratio must be kept small. However, smaller splitting ratio takes more number of iterations to balance the load. To overcome this constrain in this study, an enhanced variable splitting ratio based algorithm was proposed to reduce the delay without oscillations. The splitting ratio was varied depending upon the difference between the maximum path load and the minimum path load. A numerical method was used to evaluate the performance of the algorithm and it was applied to a test network. The simulation results showed that the number of iterations required to balance the load was decreased significantly with out oscillations.

Keywords: Load balancing, splitting ratio, delay, oscillations

INTRODUCTION

Traffic management and control in high-speed networks have become increasingly important as the Internet evolving towards an universal and convergent network capable of flowing multi-service traffic (voice, data and video) over the same IP based infrastructure. Due to the specific characteristics of each type of traffic, the network must treat them in a differentiated way in order to provide the Quality of Service (QoS) demanded by users. In this context, Multi-Protocol Label Switching (MPLS)[1-3] plays a vital role, contributing efficient Traffic Engineering (TE), high speed, QoS and optimized resource allocation by balancing the load. Distributing the load among varies Label Switched Paths (LSPs) in MPLS networks in a balanced manner is very important to utilize the available resources effectively. This fact may stimulate service providers to improve network planning techniques to adequately provide network resources.

Various load balancing algorithms are available to balance the load in MPLS networks. In static version of the load balancing methods[4], the traffic matrix is assumed to be known. However, if such information is not available or traffic condition changes unexpectedly, another approach is needed. In dynamic version of the load balancing problem, the traffic is split dynamically at flow level into multiple paths. Elwalid[5] introduced a mechanism called MPLS Adaptive Traffic Engineering (MATE). MATE is a state-dependent traffic engineering mechanism, in which the traffic load is balanced using a distributed adaptive algorithm. The algorithm tries to equalize congestion measures among the Label Switched Paths (LSPs) by approximating the gradient-projection algorithm, which transfers traffic toward the direction of the gradient projection of the objective function.

Song[6] introduced a concept called Load Distribution over Multipath (LDM), in which traffic is split dynamically at flow level into multiple paths. The LSPs for the incoming traffic is selected, randomly based on congestion and the length of the path. Here, the traffic engineering is based on the measured traffic conditions between each egress and ingress pair. Murugesan[7] introduced an Adaptive Granularity Algorithm, in which the granularity is varied adaptively to change the splitting ratio in order to reduce the oscillation and delay to reach the balanced condition. In research[8], a traffic control technique based on nonlinear control theory was proposed. There are some shortcomings in the above algorithms. In[5], traffic engineering is based on the measured traffic conditions between each ingress and egress pair. However, these measures offer overlapping information since different

Corresponding Author: G. Murugesan, Kongu Engineering College, Perundurai, Erode-638052, Tamilnadu, India

LSPs might use the same links. The amount of control data can be reduced by using only link load information. The splitting step size will produce oscillations for larger step size or requires more number of iterations for minor step size. Gradient projection method has the unavoidable inherent disadvantages, namely high complexity and slow convergence. In[6] whether the granularity of the traffic splitting at the flow level is fine enough to provide stable network conditions is not clear. In[7] the variation in the splitting ratio is depending upon the change in the traffic demand. Number of iteration required to reach the balanced condition is reduced only for smaller traffic changes. However, for the large dynamic change in the traffic demand requires more number of iterations to reach the balanced level without oscillations. Large number of iterations increases the delay. In research[8,9], there were too many assumptions and it is very difficult to obtain the delay contributed by the cross traffic which is used to calculate how much traffic should be shifted among the LSPs. To overcome the shortcomings, in this research an enhanced variable splitting ratio algorithm has been developed and simulated.

MATERIALS AND METHODS

MPLS stands for Multi Protocol Label Switching. Multi Protocol is a technique that is applicable to any network layer protocol. Label switching because switching is done using simple labels. Multilayer switching describes the integration of Layer 2 (data link layer) switching and Layer 3 (network layer) routing techniques. MPLS network consists of a set of nodes, called Label Switched Routers (LSRs) that are capable of switching and routing packets on the basis of a label which has been appended to each packet. Labels define a flow of packets between two endpoints or a multicast group of destination endpoints. For each distinct flow, called a Forwarding Equivalence Class (FEC), a specific path through the network of LSRs is defined. LSRs do not need to examine or process the IP header, but rather simply forward each packet based on its label value. The first router in MPLS domain is called as ingress router and the last router in MPLS domain is called as egress router. Constrain Based Label Distribution protocol (CR-LDP) or any one of the existing layer 3 protocol such as OSPF/IS-IS or RSVP-TE is used to form the Label Switched Paths (LSPs) between sender and receiver.

Load balancing: Load balancing is dividing the amount of work that a node has to do between two or more nodes so that more work gets done in the same amount of time and, in general, all users get served faster. In IP networks MPLS enables splitting the traffic into several paths instead of single path. Explicit routing and traffic splitting gives the ability to balance load and thus improve the performance of the network. Static load balancing is presented in [2]. The static problem is possible to be formulated and solved only if we have precise information on all the traffic demands. However, such information is not available or traffic condition change unexpectedly. In dynamic load balancing, dynamically changing network status information are utilized[5-9]. Traffic Engineering (TE) of dynamic methodologies is classified into two basic types: time-dependent and state-dependent. In time-dependent TE, traffic control algorithms are used to optimize network resource utilization in response to long time scale traffic variations. In state-dependent TE, traffic control algorithms adapt to relatively fast network state changes. State-dependent Load balancing is a key technique for improving the performance and scalability of the Internet. The fundamental problem in dynamic load balancing and job scheduling in parallel and distributed nodes involves moving load between nodes. Each node can transfer load to at most one neighbor also, any amount of load can be moved along a communication link between two nodes in one step. The dynamic load balancing is formed with incomplete information. More precisely, it is assumed that the traffic demands are unknown but the link loads are periodically measured using a measurement system as in[10,11].

Adaptive splitting ratio based load balancing algorithm: In adaptive splitting ratio based load balancing algorithm[7] load is balanced by iterative method. In every iteration, a part of the load called splitting ratio (δ) is split from the maximum cost path to the minimum cost path. The number of iterations required to balance the load among all the paths between an ingress-egress (IE) pair k decides the delay of the system to balance the load. Higher value of δ provides lower delay than smaller value. However, higher value of δ creates oscillations while splitting the load for balancing. Considering the delay and oscillations, in Adaptive Splitting Ratio based Load Balancing Algorithm, the splitting ratio is adaptively changed depending upon the difference between the calculated balanced cost and the maximum path cost by changing the granularity. Consider a network consisting

of nodes $n \in N$ and links $l \in L$. Let I_k denotes the ingress node and E_k the egress node of IE pair k. Let y_l (i) denote the measured load of link l at time i and M_l denote the capacity of the link l. The network is loaded by traffic demand d_k between IE pair $k \in K$. Each IE-pair k has a predefined set P_k of LSP's. Let $p \in P_k$ denotes paths and g denotes granularity constant. Each path $p \in P_k$ consists of a concatenation of consecutive links from I_k to E_k. Notation $l \in p$ is used to represent the link belongs to path p. Consider the measured load y_l (i) at time i are distributed to all edge routers and the decisions concerning the splitting ratios δ_p (i) can be done in a distributed way. Let path-load (ϕ_p) represents a part of the total load at the ingress node I_k of an ingress-egress pair (IE), k, distributed to the path p.

Thus, in distributed algorithm, each edge router independently determines the splitting ratios δ_p (i) for all those paths p for which it is the ingress node. Consider one such IE-pair, k. As the algorithm is adaptive, the splitting ratios will depend upon the measured link loads y_l (i). More precisely, they will depend on the estimated path costs D_p (y(i)), where y (i) = (y_l (i); $l \in L$) denotes the vector of measured loads. The objective in the adaptive granularity based distributed load balancing is to minimize the maximum cost, such as the maximum utilization. Let C_l denotes cost of link l. Then cost of a path is estimated by the cost of a link which is loaded maximum in the path between IE pair k.

$$D_p(y(i)) = \max_{l \in p} C_l(y_l(i)) \quad (1)$$

The link utilization is defined as link cost function. Link cost at time i is measured as the ratio between the measured load of the link y_l and capacity of the link M_l.

$$C_l(y_i) = \frac{y_l}{M_l} \quad (2)$$

The idea is simply to alleviate the congestion on the most costly path (among the paths belonging to the same P_k) by splitting a part its load to the least costly path. At time i, after receiving the information concerning all the measured loads y_l (i) and the difference between the balanced cost and maximum measured path cost, b, the edge router I_k finds the splitting ratio δ_p (i) as:

$$\delta_p(i) = \left(1/bg\right)\phi_q(i-1) \quad (3)$$

Then, I_k finds a path $q \in P_k$ with maximum cost and decreases its path-load as

$$\phi_q(i) = \phi_q(i-1) - \delta_p(i) \quad (4)$$

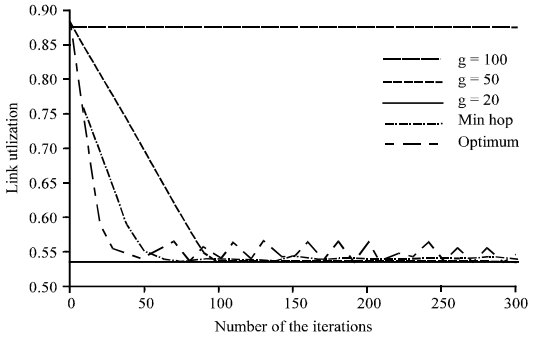

Fig. 1: Link utilization as a function of the number of iterations

Where, ϕ_q (i-1) is the path load of the path p at the previous iteration (i-1), g is a granularity constant can be set according to the delay and oscillations. The product bg is the granularity product. The variable b varies the granularity product adaptively depending upon the difference between the balanced cost and maximum measured path cost. Then, the ingress router I_k finds a path $r \in P_k$ with minimum cost and increases its path-load as:

$$\phi_r(i) = \phi_r(i-1) + \delta_p(i) \quad (5)$$

The granularity product bg and thus, the splitting ratio are adaptively varied depending upon the link load. For a small change in the link load, the granularity product is decreased and the splitting ratio is increased to reduce the number of iterations and delay to reach the steady state value. In case of a large change in the link load, the granularity product value is increased to decrease the spitting ratio and the number of iterations to balance the load become is increased.

Figure 1 depicts the link utilization versus number of iteration. For higher value of granularity constant, the splitting ratio (δ) is low enough to avoid oscillations. But, in the low δ case, the number of iterations required to balance the load is high and obviously it increases the delay to reach the balanced condition. Higher value of δ reduces the number of iterations however, oscillations are outstanding. In minimum hop routing, all the incoming traffics are routed through a shortest path at the ingress router and thus, the shortest path gets maximum load. Optimum value is calculated using static load balancing method by considering the traffic at the ingress router is known and equally distributing it to all paths in the IE pair k.

Figure 2 shows the delay analysis of adaptive granularity algorithm. The difference between the

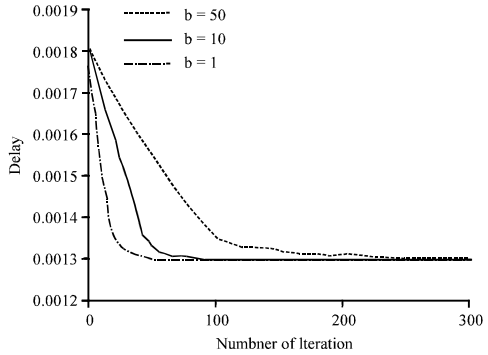

Fig. 2: Total mean delay in adaptive Algorithm

minimum measured path cost and maximum measured path cost, b, is taken in the form of percentage. In an unbalanced network, some of the links will be overloaded and the excess data than the capacity is kept in queue. This increases the delay to reach the destination. In a well balanced network, the delay. Figure 2 shows that the delay (in ms) is at a maximum value at the 0th iteration because of total unbalanced state and the delay is at a minimum value at fully balanced condition. Comparing to Fig. 1 and 2, the adaptive granularity algorithm (Fig. 2) reduces the number of iterations required to reach the balanced state without oscillation for a smaller change in the dynamic link load. However, for the large variations in traffic, the granularity product value is increased to reduce the splitting ratio and avoid oscillations. In this case the number of iteration and the delay required to balance the load is increased. Practically, the traffic varies dynamically and the amount of variation also entirely random. Hence a modified version of algorithm is required to reduce the number of iterations to reach the balanced state without oscillations for any amount of traffic variation.

Enhanced variable splitting ratio based load balancing algorithm: In enhanced variable splitting ratio based load balancing algorithm the cost of a highest loaded link in a path p is taken as the path cost and it is calculated using the Eq. (2). A part of the load is split from the maximum cost path to the minimum cost path. The splitting ratio is varied adaptively by a parameter called net granularity product. In adaptive splitting ratio based load balancing algorithm[7], the number of iteration is depending upon the change in the change in link load from the average value, that is the difference between the balanced cost and the maximum measured path cost, b. Number of iteration required to balance the load is comparatively less for a lower b value than at higher value of b. Hence at a higher value of b, the delay to balance the load with out oscillations is again become higher value as in the case of the algorithm without the parameter b. Hence a new effective algorithm is required to reduce the number of iterations required to balance the load even for a higher dynamic change in the link load.

Enhanced variable splitting ratio based load balancing algorithm has been developed by considering the minimum measured link cost a_i at time i, maximum measured link cost m_i of the paths p between the IE pair k at time i, full load cost of the link f_l and the number of paths P between the IE pair k. When the difference between the minimum link cost a_i and the maximum measured link cost m_i is a higher value then, the δ_p (i) will also be kept at higher value than the case of lower difference value. This is incorporated in the Eq. (3) by introducing a new parameter called net granularity product T instead of the granularity product bg as:

$$T = \frac{\left[f_l - (m_i - a_i)\right]}{x} + P \qquad (6)$$

The minimum value of T is equal to P. This is true if the difference between the a_i and m_i is equal to the full load cost f_l. This represents an initial condition. Before load balancing, all the traffic is sent only through the shortest path and the shortest path will be fully loaded and (m_i-a_i) is at maximum. In this initial condition, the total load is divided in to P parts to split into the minimum loaded path. The parameter x is a trade of parameter and its value is set according to the allowed magnitude of oscillations and delay. Increasing the value of x will decrease the magnitude of oscillations. But, higher value of x will increase the delay. Comparing to the Adaptive Splitting Ratio based Load Balancing Algorithm; the delay produced by the trade-off factor x is very less. The difference between the a_i and the m_i is reduced in successive iterations by splitting the traffic from m_i to a_i. When the difference reaches a lower value called threshold value (t), the value of net granularity product is increased to Tg to reduce the splitting ratio. The value of granularity constant is set by considering the delay and the magnitude of oscillations. The Tg will reduce the δ_p and also will reduce the oscillations when the difference between a_i and m_i reaches the threshold value. For example, if t = 0.05, the net granularity product is set to T up to the difference between a_i and m_i reaches the value of 5% of the full load path cost. After threshold the net granularity product is set to Tg. The algorithm is executed at all the ingress routers of MPLS network operates as follows:

- Calculate link cost of each links using Eq. 2
- Calculate the path cost of all the paths using Eq. 1
- Find a path $q \in P_k$ with maximum cost and a path $r \in P_k$ with minimum cost
- Calculate the net granularity product T using the Eq. (6)
- If $(m_i-a_i)>t$ then, calculate the splitting ratio $\delta_p(i)$ as:

$$\delta_p(i) = \left(1/T\right)\phi_q(i-1) \quad (7)$$

- Decrease the path-load of the path q as:

$$\phi_q(i) = \phi_q(i-1) - \delta_p(i) \quad (8)$$

and increase the path-load of the path r as

$$\phi_r(i) = \phi_r(i-1) + \delta_p(i) \quad (9)$$

- For the other entire paths keep the old splitting ratio
- Else if $(m_i-a_i)\leq t$ then, calculate the splitting ratio δ_{pt} (i) as:

$$\delta_{pt}(i) = \left(1/Tg\right)\phi_q(i-1) \quad (10)$$

- Decrease the splitting ratio of the path q as:

$$\phi_q(i) = \phi_q(i-1) - \delta_{pt}(i) \quad (11)$$

and increase the splitting ratio of the path r as:

$$\phi_r(i) = \phi_r(i-1) + \delta_{pt}(i) \quad (12)$$

- For the other entire path keep the old splitting ratio.

Performance evaluation: The evaluation method is iterative and runs as follows: Consider a test network that includes the nodes n, links l, IE-pairs k and paths p. The traffic demands d_k are first fixed. Path-load ϕ_p is initiated by allocating the traffic demands to the paths with the minimum hop-count. At each iteration i, the link loads y_l (i) induced by the path-load ϕ_p (i-1) are calculated by:

$$y_l(i) = \sum_{k\in K} \sum_{p\in P_k : l\in p} d_k\phi_p(i-1) \quad (13)$$

The link l may be a common link for more than one ingress-egress pair $k \in K$ and $p \in P_k$. Total load is calculated using the Eq. (13).

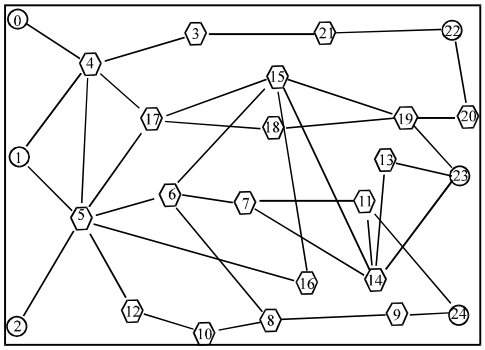

Fig. 3: Simulated network topology

Figure 3 was the test network considered for the evaluation. It consists of 24 nodes and 36 links. The number of possible paths was limited by requiring that the paths $p \in P_k$ do not have any common links. All hexagon nodes are MPLS enabled nodes and circle nodes are Non MPLS nodes. Link capacity of all the nodes was considered equal. Before load balancing some of the links are over utilized than other links. This is because of minimum hop count path selection mechanism. Load balancing is achieved in successive iterations. The performance of the algorithm is analyzed with minimum link utilization (a_i), the average link utilization (a_v), the maximum link utilization (m_i) and trade-off parameter x. In Fig.3 three Ingress-Egress (IE) pair nodes 4-14, 5-19 and 3-20 were considered for testing. Before load balancing, as the link between the nodes 15 and 17 was a common link for shortest paths of all the three IE pairs, it was over loaded. Load balancing algorithm at ingress nodes of all the three IE pairs split a part of load which is equal to splitting ratio from the maximum loaded path to the minimum loaded path.

RESULTS AND DISCUSSION

The adaptive splitting ratio based load balancing algorithm[7] reduces the number of iterations with very less magnitude of oscillations for small variations in the link load but, not suitable for higher link load variations. In Fig. 1, more than 50 iterations are required to balance the load without oscillations for various link load conditions and in Fig. 2, number of iterations is reduced to 25 for smaller link load variations but, it is increased up to 100 iterations for large link load variations. In case of effective variable splitting ratio based load balancing algorithm, the number of iteration required to balance the load without oscillation is less than 11 for the different value of link load conditions.

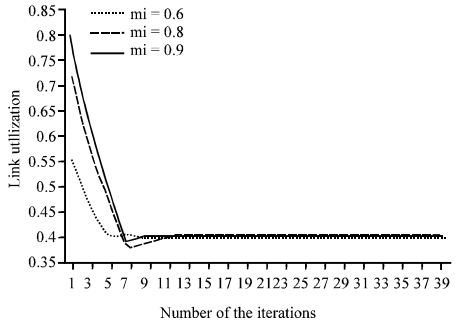

Fig. 4: Link utilization with a_v = 0.4 (40%)

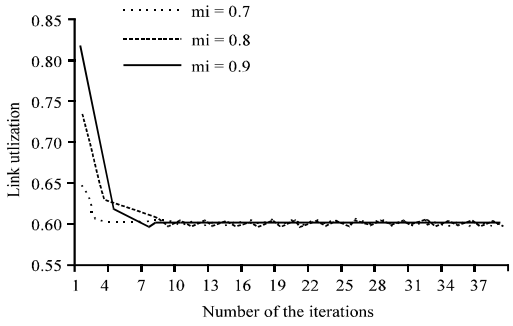

Fig. 5: Link utilization with a_v = 0.6

Load was applied to the test network such a way that the average link utilization a_v = 0.4 (40% of the maximum path cost) and m_i = 0.6. The link load of the maximum loaded link was measured in successive iteration. It was repeated for m_i = 0.8 and m_i = 0.9. Figure 4 depicts the link utilization as a function of number of iterations with x = 6, t = 0.01 and g = 10. Link utilization is at a maximum value at 0th iteration. This link cost is reduced by splitting the load to the minimum cost path into minimum cost path in successive iterations. It is interested to note that the number of iterations required to reach the average value is approximately equal for different values of m_i (60, 80 and 90%). If the minimum link utilization itself at a higher value, the available free bandwidth to accommodate the new load will be less. At this situation, as per the Eq. (7), the value of path-load ϕ_q (i-1) is high and the δ that is to be split from the highest cost path to lowest cost path also becomes high. This higher value of splitting ratio generates oscillations at the balanced state. Figure 5 illustrates the link utilization as a function of number of iterations with a_v = 0.6 (60%). As all the links are utilized nearing to the maximum value, the magnitude of oscillations is increased with increasing value of a_v.

Figure 6 illustrates the link utilization of a maximum cost link with a_v = 0.8 (80%) and m_i = 85, 90 and 95%. Independent of the value of m_i, oscillations are outstanding because of high splitting ratio at a_v = 0.8. To avoid the oscillations, the net granularity product has to be increased to reduce the splitting ratio. The trade-off parameter x can be used for this purpose. To avoid oscillations x value has to be decreased to decrease the splitting ratio.

Figure 7 depicts the result of the effective variable splitting ratio based algorithm with a_v = 60%, trade-off parameter x = 4, t = 0.02 and maximum link cost m_i = 0.8 (80%). The link utilization of a highest cost link is reduced from 80-62% in 10 iterations because of higher

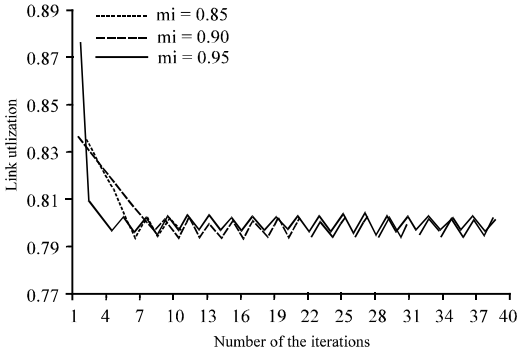

Fig. 6: Link utilization with a_v = 0.8

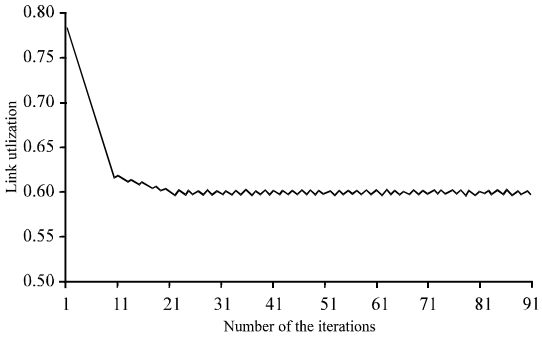

Fig. 7: Link utilization with a_v = 0.6 and x = 4

splitting ratio. Thus, load balancing reaches the near balanced state in 10 iterations. After reaching 62% of link utilization, to avoid oscillations, the net granularity product is increased to Tg and thus the splitting ratio δ is reduced from the Eq. 7 to the Eq. (10) and 10 more iterations are required to reach the average value of 60%. As per the Fig. 1 and 2, the number of iterations required to balance the load for a small variations in the link load was more than 25 and for large variations, it was 50 and more. Figure 4 and 7 shows that, even for large variations in the link load, the enhanced variable

splitting ratio based load balancing algorithm reduces the number of iterations to 20. The results show that the proposed algorithm gives significant improvement in the number iterations required for balancing the load in MPLS networks.

CONCLUSION

Enhanced variable splitting ratio algorithm was presented to balance the load effectively in the MPLS enabled IP networks. The results showed that the number of iterations required to balance the load without oscillations was reduced. Comparing the Fig. 1, 2, 4 and 7, the maximum number of iterations required to balance the load for different values of link load was reduced from 50-20. Because of fast load balancing, this algorithm supports rerouting in case of any link failure by balancing the rerouted load with very less delay.

REFERENCES

1. Uyless Black, 2002. MPLS and Label Switching Networks. 2nd Edn., Pearson Education (Singapore) Pte. Ltd, Indian Branch, Delhi, pp: 69-101 and 162-191.
2. Grenville Armitage, 2000. MPLS the magic behind the myths. IEEE Commun. Mag., 38: 124-131.
3. Rosen, E., A. Viswanathan and R. Callon, January, 2001. Multiprotocol Label Switching Architecture. Internet Engineering Task Force, RFC3031. http://www.ietf.org/rfc/rfc3031.txt
4. Bertsekas, D. and R. Gallager, 1992. Data Networks. 2nd Edn., Prentice-Hall. Prentice-Hall, Inc. Upper Saddle River, NJ, USA
5. Elwalid, A., C. Jin, S. Low and I. Widjaja, 2001. MATE: MPLS adaptive traffic engineering. IEEE Infocom., Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, Volume-3, April 22-26 2001, Anchorage, Alaska, USA. ISBN 0-7803-7016-3, 1300-1309. http://www.ieee-infocom.org/2001 DOI: 10.1109/INFCOM.2001.916625.
6. Song, J., S. Kim, M. Lee, H. Lee and T. Suda, 2003. Adaptive load distribution over multipath in MPLS networks. IEEE International Conference on Communications (ICC'03), Anchorage, Alaska, Date: 11-15 May 2003 pp: 233-237. DOI:10.1109/ICC.2003.1204176 www.icc2003.com
7. Murugesan, G. and A.M. Natarajan, 2007. Adaptive granularity algorithm for effective distributed load balancing and implementation in multiprotocol label switching networks. IEEE, International Conference on Advanced Computing and Communication (ADCOM'07) 18-21 December 2007, pp: 626-631. DOI:10.11.09/ADCOM.2007.84 www.iitg.ac.in/adcom07
8. Dinan, E., D. Awduche and B. Jabbari, 2000. Analytical framework for dynamic traffic partitioning in MPLS networks IEEE International Conference on Communications, (ICC'00), 18-22 June 2000, New Orleans, Volume-3, pp: 1604-1608. DOI:10.1109/ICC.2000.853766, http://ieeexplore.ieee.org/xpl/RecentCon.jsp?punumber=6882
9. Dengyin Zhang, Zhiyun Tang and Ruchuan Wang, 2006. Automatic traffic balance algorithm based on traffic engineering. J. Network Syst. Manage., 14: 317-325.
10. Butenweg, S., 2003. Two distributed reactive MPLS traffic engineering mechanism for throughput optimization in Best effort MPLS networks, Eigth IEEE International Symposium on Computers and Communications (ISCC'03). 379-384 vol.1, 30th June-3rd July 2003, KEMER - ANTALYA, TURKEY, Digital Object Identifier 10.1109/ISCC.2003.1214149 http://www.comsoc.org/iscc/2003
11. Ashwin Sridharan, Roch Guerin and Christophe Diot, 2005. Achieving near-optimal traffic engineering solution for current OSPF/IS-IS networks. IEEE/ACM Trans. Network., 13: 234-247.