LOAD BALANCING IN MPLS

Prabhjot Kaur, Satish Devane
Datta Meghe College of Engineering, Navi Mumbai, Maharashtra, India

ABSTRACT
MPLS is a mechanism in high-performance telecommunications network that directs data from one network node to the next based on short path labels rather than long network addresses, avoiding complex lookups in a routing table. Traffic Engineering (TE) is the network engineering technology which makes a business flow mapped to the actual physical access, but also can automatically optimize the network resources to achieve specific performance requirements of application services, with macro-regulation and micro-control. MPLS and MPLS-TE are playing a key role in resource management of contemporary packet networks and also in the provisioning of end-to-end virtualized network services. In this context, route planning and resource allocation in MPLS networks, conforming to MPLS-TE requirements, is of vital importance for optimizing network usage. One of its key objectives is to balance loads across a network, i.e. Load Balancing is an important aspect of Traffic Engineering. It can use some mechanisms to map a part of traffic in the over utilized routes to some underutilized routes to avoid congestion in the Shortest Path route, and to promote total network throughput and network resource utilization MPLS has many great advantages in supporting Traffic Engineering. In this paper we are considering various algorithms which are related to load balancing in MPLS with no priority conditions and also few algorithms used for priority conditions and multicast traffic and their analysis is provided based on ns2 simulator.

INDEX TERMS: *QOS, CR-LSP, ER-LDP, FEC, LSR, Route-ID, HOP.*

1. INTRODUCTION

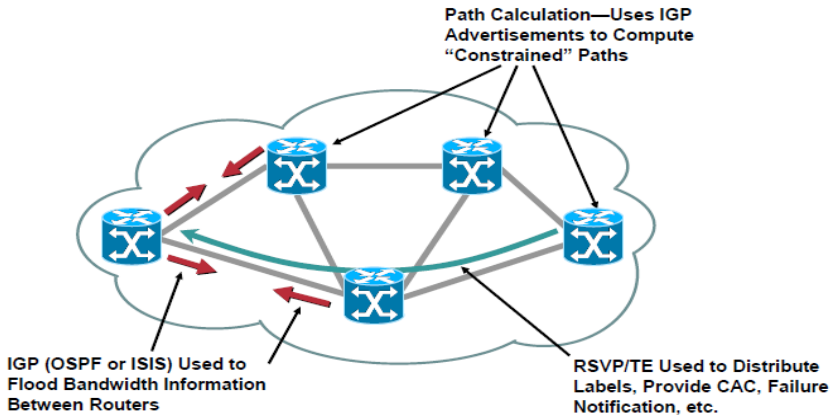
Traffic Engineering (TE) is the network engineering technology which makes a business flow mapped to the actual physical access, but also can automatically optimize the network resources to achieve specific performance requirements of application services, with macro-regulation and micro-control.
The main purpose of traffic engineering is to promote the effective and reliable network operations, at the same time, optimizing the utilization of network resources to improve network performance. Traffic engineering solves QoS descend problems that result from the business flow between available resources and the mapping efficiency is not high from optimization and meeting restriction, simply made up of inadequate of integrated services (Intserv) and differentiated services (Diffserv) two QOS model at the macro aspects of the use of network resources.
MPLS and MPLS-TE are playing a key role in resource management of contemporary packet networks and also in the provisioning of end-to-end virtualized network services. In this context, route planning and resource allocation in MPLS networks, conforming to MPLS-TE requirements, is of vital importance for optimizing network usage.
Load balancing is a key aspect of MPLS-TE refers to the process of adjusting the workload on a collection of systems or applications to ensure that no single system or application is under or over utilized. It requires an ability to control traffic flows precisely. Uses some mechanisms to map a part of traffic in the over utilized routes to some underutilized routes to avoid congestion in the Shortest Path route. It allows efficient explicit routing of LSP, which can be used to optimize the utilization of network resources and enhance traffic oriented performance characteristics.

MPLS can use Constraint-based Routing Label Distribution Protocol (CR-LDP) [12] to establish an ER-LSP, and map part of a traffic flow to this ER-LSP. Because Explicitly-Routed Label Switched Path (ER-LSP) can select a route that not restricted in Shortest-Path route, we can use an ER-LSP to map a part of traffic flow to some under utilized links when congestion happens at Shortest-Path route, so as to reduce congestion and simultaneously enhance network resource utilization.
In this paper, for the goal of study simply, we assume that one ER-LSP bears only one Traffic flow. It is to say that each ER-LSP carry a traffic flow. Under condition that all traffic flows have different priorities, higher priority ER-LSP can preempt the resource of lower priority ER-LSP to guarantee QoS of higher priority service; under no priority condition, one ER-LSP can't preempt other ER-LSP's resource. When there are more than one parallel routes connect two nodes, the task for Load Balancing is to map traffic flows to these routes to get the maximal network throughput and simultaneously enhance the total network resource utilization. In this paper, we present three Load-Balancing algorithms for non-priority unicast traffic. Simulation results for these three load balancing algorithms are analyzed in detail and also introduction to algorithms based on multicast traffic and with congestion control are provided.

2. MPLS-TE

Multiprotocol Label Switching (MPLS) is a mechanism in high-performance telecommunications networks that directs data from one network node to the next based on short path labels rather than long network addresses, avoiding complex lookups in a routing table. The labels identify virtual links (paths) between distant nodes rather than endpoints. MPLS can encapsulate packets of various network protocols. MPLS supports a range of access technologies, including T1/E1, ATM, Frame Relay, and DSL. It is a highly scalable, protocol agnostic, data-carrying mechanism. In an MPLS network, data packets are assigned labels. Packet-forwarding decisions are made solely on the contents of this label, without the need to examine the packet itself. This allows one to create end-to-end circuits across any type of transport medium, using any protocol. The primary benefit is to eliminate dependence on a particular OSI model data link layer technology, such as Asynchronous Transfer Mode (ATM), Frame Relay, Synchronous Optical Networking (SONET) or Ethernet, and eliminate the need for multiple layer-2 networks to satisfy different types of traffic. MPLS belongs to the family of packet-switched networks.
MPLS makes it easy to commit network resources in such a way as to balance the load in the face of a given demand and to commit to differential levels of support to meet various user traffic requirements. The ability to dynamically define routes, plan resource commitments on the basis of known demand, and optimize network utilization is referred to as traffic engineering.

TE FUNDAMENTALS: "Building Blocks"



The process of routing data traffic in order to balance the traffic load on the various links, routers, and switches in the network is called MPLS-TE. It is kind of a key in most networks where multiple parallel or alternate paths are available. It also enables MPLS backbone to replicate and expand upon the traffic engineering capabilities of Layer 2 ATM and Frame Relay networks.

Traffic engineering is essential for service provider and Internet service provider (ISP) backbones. Such backbones must support a high use of transmission capacity, and the networks must be very resilient, so that they can withstand link or node failures. MPLS traffic engineering provides an integrated approach to traffic engineering. With MPLS, traffic engineering capabilities are integrated into Layer 3, which optimizes the routing of IP traffic, given the constraints imposed by backbone capacity and topology.

3. NEED OF TRAFFIC ENGINEERING

1) Congestion in the network due to changing traffic patterns.
2) Better utilization of available bandwidth
3) Route around failed links/nodes.
4) Build new services—virtual leased line services
5) Capacity planning.

4. LOAD BALANCING

Load balancing is an even division of processing work between two or more computers and/or CPUs, network links, storage devices or other devices, ultimately delivering faster service with higher efficiency. In simple words Load balancing is the process by which inbound Internet protocol (IP) traffic can be distributed across multiple servers. This enhances the performance of the servers, leads to their optimal use, and ensures that no single server is overwhelmed, is accomplished through software, hardware or both, and it often uses multiple servers that appear to be a single computer system (also known as computer clustering).

Typically, two or more web serves are employed in a load balancing scheme. In case one of the servers begins to get overloaded, the requests are forwarded to another server. This process brings down the service time by allowing multiple servers to handle the requests. Service time is reduced by using a load balancer to identify which server has the appropriate availability to receive the traffic. The process, very generally, is straightforward. A webpage request is sent to the load balancer, which forwards the request to one of the servers. That server responds back to the balancer, which in turn sends the request on to the end user. Load balancing allows service to continue even in the face of server down time due to server failure or server maintenance.

Assigning a Load Factor: *Load* refers to a number assigned to a service request based on the amount of time required to execute that service. Loads are assigned to services so that the system can understand the relationship between requests. To keep track of the amount of work, or total load, being performed by each server in a configuration, the administrator assigns a *load factor* to every service and service request.

A load factor is a number indicating the amount of time needed to execute a service or a request. On the basis of these numbers, statistics are generated for each server and maintained on the bulletin board on each machine. Each bulletin board keeps track of the cumulative load associated with each server, so that when all servers are busy, the system can select the one with the lightest load.

To determine how to assign load factors (in the SERVICES section of UBBCONFIG), run an application for a long period of time and note the average time it takes to perform each service. Assign a LOAD value of 50 (LOAD=50) to any service that takes roughly the average amount of time. Any service taking longer than average should have a LOAD>50; any service taking less than the average should have a LOAD<50.

5. LOAD BALANCING IN MPLS

When multiple TE tunnels have the same cost, traffic can be load-balanced across them. Traffic can also be load-balanced between the native IP path and TE tunnels if the cost of the routing is the same. This situation has some restrictions. When you are load balancing over TE tunnels, the load balancing can even be unequal cost load balancing.

The load balancing of traffic is weighted proportionally to the bandwidth requirement of the TE tunnels. If you have one tunnel with 80 MB and one with 20 MB of reserved bandwidth, the load-

balancing ratio is 4:1, or the first tunnel should get four times more traffic than the second tunnel. However, the load-balancing ratio is an approximation, because Cisco Express Forwarding (CEF) has only 16 hash buckets.
When an LSR performs the load balancing over one or more IP paths and one or more TE tunnels, it is always equal cost load balancing. This means that every path gets the same amount of traffic. Multiple TE tunnels can be handy when the amount of bandwidth to be reserved between a pair of routers is more than the bandwidth capacity of the links. You can then just create multiple TE tunnels with each a piece of the required bandwidth.

Need Load balancing in MPLS

1) TE has become an essential requirement for the ISPs to optimize the utilization of existing network resources and to maintain a desired overall QoS with fewer network resources.
2) MPLS has many great advantages in supporting Traffic Engineering.
3) Load balancing is a key aspect of MPLS TE, it requires an ability to control traffic flows precisely.
4) Uses some mechanisms to map a part of traffic in the over utilized routes to some underutilized routes to avoid congestion in the Shortest Path route.
5) Promote total network throughput and network resource utilization.
6) When there are more than one parallel routes connect two nodes, the task for Load Balancing is to map traffic flows to these routes to get the maximal network throughput and simultaneously enhance the total network resource utilization.
7) It allows efficient explicit routing of LSP, which can be used to optimize the utilization of network resources and enhance traffic oriented performance characteristics.
8) For example, multiple paths can be used simultaneously to improve performance from a given source to destination.

6. VARIOUS ALGORITHMS FOR LOAD BALANCING IN MPLS

Various algorithms are introduced by various researchers which are either dedicated to the no priority conditions or various priority condition or multi cast services or either global load balancing.
We here are considering the few algorithms which are related to no priority condition and there analysis will be done by a simulator to show which of the following is best and why.
Three algorithms for load balancing in MPLS Traffic Engineering are as under:

1) The Static Load Balancing Algorithms
    a) Topology-Based Static Load-Balancing Algorithm (TSLB),
    b) Resource-Based Static Load-Balancing Algorithm (RSLB) and
2) Dynamic load-balancing algorithm (DLB).

6.1 Static Load-Balancing Algorithm

(a) Topology-Based Static Load-Balancing Algorithm (TSLB)

Algorithm Description:
First, we construct a data structure named route to describe a route character, which include its resource availability and its topology character:
Route = {< Route-ID >< HOP >< Length >< Free-capacity >}
Route-ID is identification of one route.
HOP is the sequence of nodes that route goes through.
Length is the number of links that route composes of.
The meaning of **Free-capacity** is: we suppose route R composes link Li, j , here, i is the Upstream node of Li, j and j is the Downstream node of Li, j .
If the unused capacity of Li, j is Fi, j , then Free-capacity in router R is defined as MIN(Fi, j) .
After that we construct a data structure named Selectable- Route-Collection (SRC). SRC composes all parallel routes between Ingress and Egress:
SRC = {n route, route ...route 1 2}
When Ingress node receives a new traffic flow, it will establish SRC first, and then go to (1):

(1) If SRC is NULL, then go to (4); else go to (2);
(2) Select the route whose Length is **shortest** in SRC, if its Free-capacity is larger than the bandwidth request of the new traffic flow, go to (5); else go to (3);
(3) Delete the shortest route from SRC; go to (1);
(4) Capacity is insufficient to satisfy the bandwidth request of the new traffic flow, it fails to transport the new traffic flow, and algorithm ends.
(5) Establish a new ER-LSP along the selected route and map the new traffic flow to this ER-LSP, after that update this route's Free-capacity, and algorithm finishes.

(b) Resource-Based Static Load-Balancing Algorithm

Algorithm Description:
The data structures defined in this algorithm are the same as that defined in TSLB. When Ingress receives a new traffic flow, it will establish SRC first, and then go to (1).
(1) If SRC is NULL, then go to (4), else go to (2).
(2) Select the route who's Free-capacity is the smallest in SRC. If its Free-capacity is larger than the new traffic flow's bandwidth request, then go to (5); else go to (3).
(3) Delete the route who's Free-capacity is the smallest from SRC, and then goes to (1).
(4) Capacity is insufficient to satisfy the bandwidth request of the new traffic flow, it fails to transport the new traffic, and algorithm ends.
(5) Establish a new ER-LSP along the selected route and map the new traffic flow to this ER-LSP. After that update this route's Free-capacity, and algorithm finishes successfully.

6.2 Dynamic Load-Balancing Algorithm (DLB)

(c) Dynamic load-balancing algorithm

*Algorithm Description***:**
First, we construct two data structures named route and Selectable-Route-Collection (SRC) as the same as defined in TSLB.
After that we construct a data structure named ER-LSP to describe the resource utilization of an ER-LSP:
ER-LSP＝ {< Traffic ID >< LSPID ><Route-ID>< UB >< FC >< CB >}
Here,
Traffic ID indicates the traffic that this ER-LSP bears.
LSPID is the identification of the ER-LSP.
UB (used bandwidth) is the resource consumed by the ER-LSP.
The meaning of FC (Free Capacity) is: we suppose ER-LSP R goes through links marked Li, j, which i, j is the Upstream node or Downstream node of Li, j. If unused capacity of Li, j is Fi, j , then FC = MIN(Fi, j) , CB (Contributed Bandwidth) is the free resource available for use in common if this ER-LSP gives up its resource, it is to say that CB =UB + FC .
We now use variable pending-Traffic to keep the Traffic ID that is to select route or to reroute. For each traffic flow that is to select route or to reroute, we construct a Negotiate ER-LSP Collection (NLC). NLC composes of all parallel ER-LSPs whose UB is smaller than bandwidth request of this traffic flow and whose route is in SRC:
NLC LSP, ER − LSP = {n ER − ...ER − LSP 1 2}
Last, we construct a stack named Traffic-Stack (TS). When Ingress node receives a new traffic flow, it will establish SRC, empty TS, and then assign this traffic flow's Traffic ID to pending-Traffic, go to (1).
1) First use TSLB to select route for the traffic flow whose Traffic ID equals to the value of pending-Traffic. If success, go to (9); else go to (2)
(2) Establish SRC and establish NLC for the traffic whose Traffic ID equals to the value of pending-Traffic, go to (3);
(3) If NLC is NULL, then go to (6), else go to (4);

(4) Select the ER-LSP whose UB is the smallest in NLC. If its CB is larger than the bandwidth request of the traffic flow whose Traffic ID equals to the value of pending- Traffic, then go to (7), else go to (5);
(5) Delete the ER-LSP whose UB is the smallest in NLC, go to (3);
(6) Capacity is insufficient to satisfy the bandwidth request of the new traffic flow, it fails to transport the new traffic, and algorithm ends. Push contemporary pending-Traffic value to TS top, then assign the Traffic ID of the ER-LSP whose UB is the smallest in NLC to pending-Traffic. After that, delete the route this ER-LSP belongs to from SRC, go to (1);
(8) Release the ER-LSP whose Traffic ID equals to the value of pending-Traffic, pop TS top to pending-Traffic, go to (1);
(9) If TS is NULL, then algorithm finishes successfully; else go to (8);

7. SIMULATIONS AND EVALUATION

In this section, we use Figure (e) as the example. Under worst condition, use LBL NS-2[7] simulator to evaluate Shortest-Path algorithm and three Load Balancing algorithms proposed in this paper. The network through put and utilization for each algorithms are evaluated, Simulation results are shown as Figures (a-d)

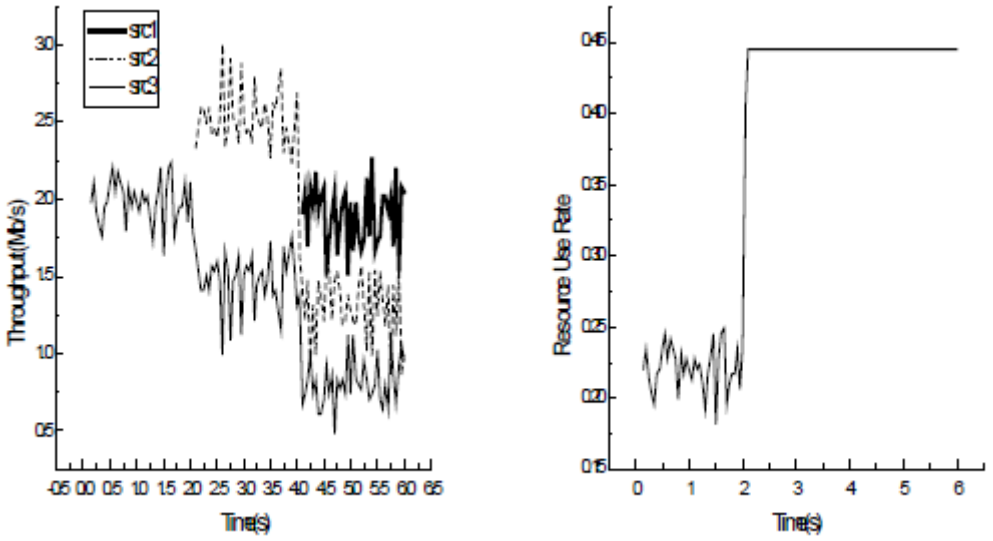


Fig (a) Throughput of each traffic flow (left) and resource utilization (right) using Shortest-Path algorithm

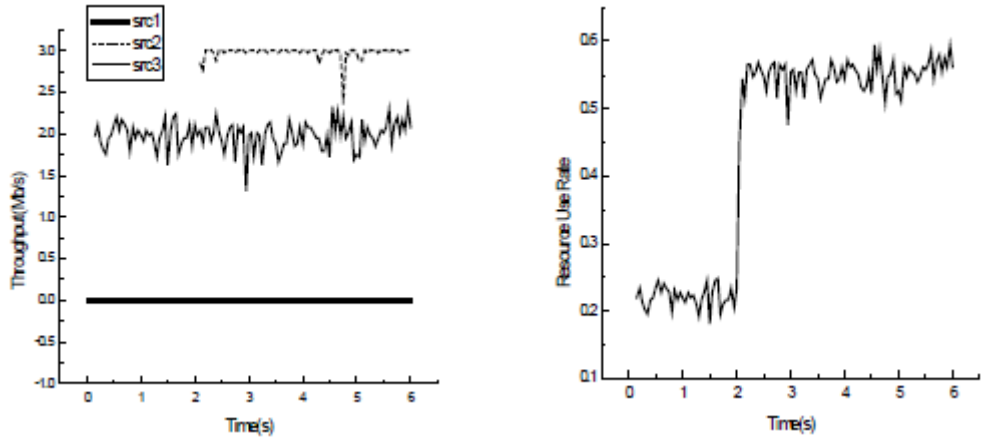


Fig (b) Throughput of each traffic flow (left) and resource utilization (right) using TSLB Algorithm

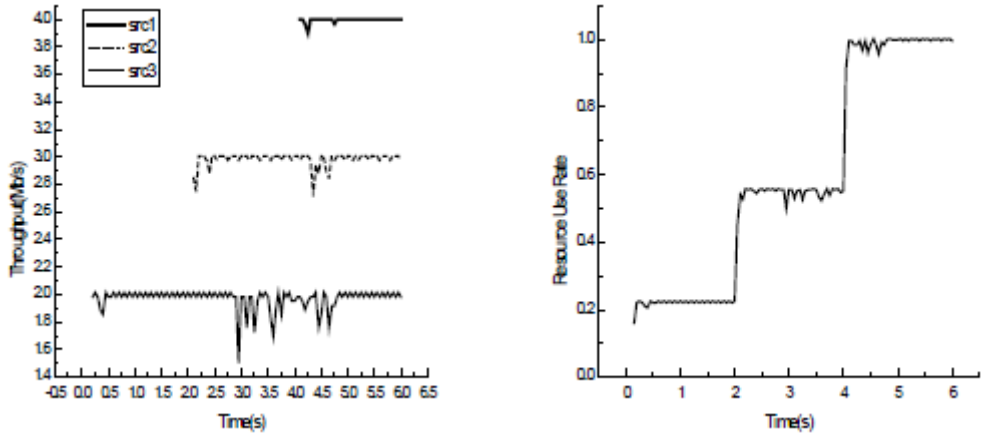


Fig (c) Throughput of each traffic flow (left) and resource utilization (right) using RSLB Algorithm

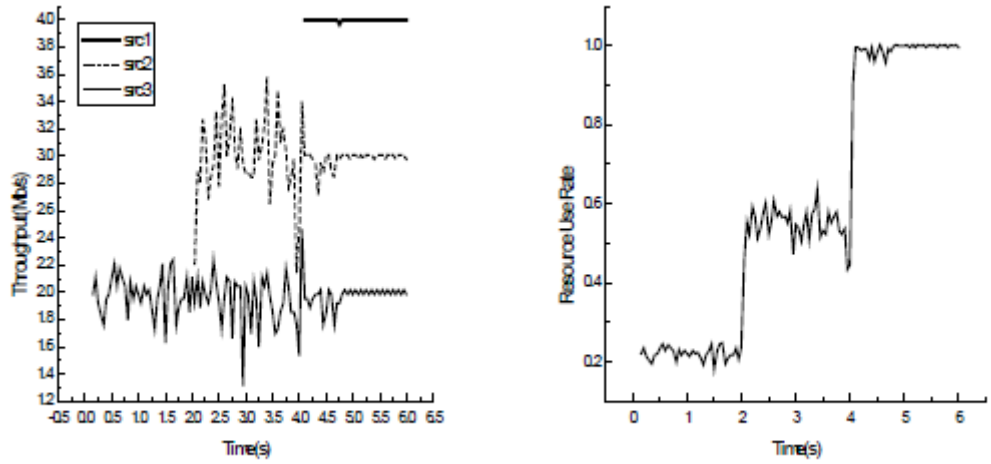


Fig (d) Throughput of each traffic flow (left) and resource utilization (right) using DLB Algorithm

Analysis of all three algorithms

1) ***Analysis of TSLB*:**

This algorithm is enhanced from Shortest-Path algorithm. In this algorithm, new traffic flow is routed through Shortest-Path first; if Shortest-Path has insufficient capacity to satisfy the bandwidth request of new traffic flow, then this traffic flow will select the longer route; If there is no route satisfies the new traffic flow's bandwidth request, algorithm fails. This algorithm reduces congestion in Shortest-Path greatly.

But this algorithm has its shortcoming and cause low utilization of network resource. As shown in Fig.6.5, there are 3traffic flows: from Src1 to Dst1, from Src2 to Dst2 and fromSrc3 to Dst3. Src1 requests average bandwidth 4Mbps and burstiness less than 5.3Mbps; Src2 requests average bandwidth 3Mbps and burstiness less than 3.7Mbps; Src3requests average bandwidth 2Mb and burstiness less than2.4Mbps. If the sequence for sources sending data is Src1 first, then Src2 and Src3 lastly, according to this algorithm, every traffic flow can map to its appropriate route. In fact, the sending sequence for sources is random. If the sending sequence is Src3 first , then Src2 and Src1 lastly, according to this algorithm, traffic flow from Src3 to Dst3 will select routeR5-R6-R11-R14; traffic flow from Src2 to Dst2 will select route R4-R6-R7-R11-R13; when Src1 began sending data, because it can't find sufficient available capacity in all 3parallel routes and it failed be transported. So we can see that under some special condition this algorithm will distribute resource unreasonably, as a result it can lead to low throughput and low network resource utilization.

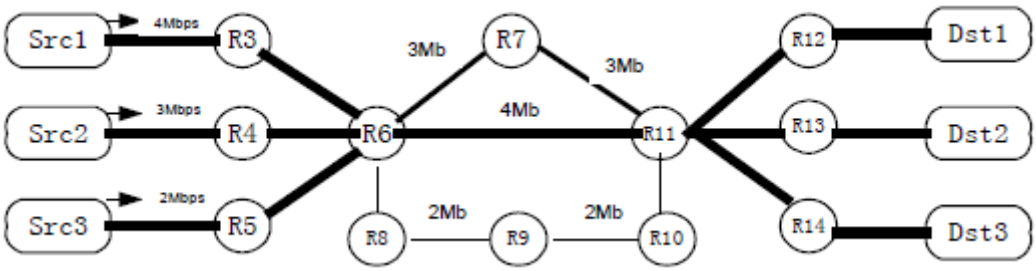


Fig (e) Network Topology with three parallel routes between Ingress and Egress

So, we propose Resource-Based Static Load-Balancing Algorithm (RSLB). It can reserve sufficient resource for future-arriving traffic flow which requests larger bandwidth.

2) ***Analysis of RSLB***:

When a new traffic flow arrives, Ingress will select a route whose Free-capacity is nearest to the traffic flow's bandwidth request (but larger than the request) in SRC. So it can reserve larger capacity route for future-arriving traffic flow which has larger bandwidth request.

But in fact, traffic appears in network is random, and each traffic flow's sending rate can be unstable; it can fluctuate in a special range. If there is only a little number of low rate traffic flows on sending for a long time, then some large capacity links will be unused for that time. At the same time, because traffic flow is mapped on a route who's Free-capacity is very near to its bandwidth request; link capacity can't fit for the traffic flow's sending rate fluctuation well, and can result in data loss in a special time when traffic flow's burst rate exceeds to the link's capacity.

The shortcoming of Static Load-Balancing Algorithms

In above section, we have presented two Static Load- Balancing Algorithms: TSLB and RSLB. These two algorithms both have their shortcomings. In some special case, they can result in low resource utilization. The fundamental reason for this is the fact that these algorithms are both static.

Under no priority condition, once an ER-LSP established, its resource can't be pre empted by other ER-LSP, and it can't reroute itself to other better route (except link error occurs).

We can imagine that under some condition, if an established ER-LSP reroute its traffic to other route so as to reserve resource for larger traffic flow, then we can enhance network throughput and resource utilization and simultaneously grantee not interrupting the rerouted ER-LSP's traffic transport. So, we present the Dynamic Load-Balancing Algorithm to realize that objective.

3) ***Analysis of DLB***:

DLB can take into account network topology character and traffic bandwidth request simultaneously. Under light load condition, traffic flows can be mapped on the short and high capacity route; when load changes to heavy, small traffic flow can reroute to another appropriate route so as to reserve high capacity route for large traffic flow. DLB algorithm can greatly enhance network throughput and resource utilization.

Among these three Load Balancing algorithms, **DLB is the best**.

DLB Algorithm complexity Analysis as below:

In DLB Algorithm, we need to establish NLC for new arriving traffic flow, and retrieve an ER-LSP whose CB is larger than this traffic flow's bandwidth request; after finding one, we need to reroute this ER-LSP. The complexity of DLB depends on the retrieve times and reroute times.

We suppose there are n ($n \geq 2$) parallel routes and there are K ER-LSPs established in common on these routes. If the new arriving traffic flow's bandwidth is larger than all established ER-LSP's UB, then its NLC composes K ER-LSPs. In the worst case, we can't find an ER-LSP whose CB is larger than this traffic flow's bandwidth request until we have retrieved all K ER-LSPs. If we fail in using TSLB to reroute this ER-LSP, then we must repeat above-mentioned procedure.

In the worst case, that procedure needs to repeat n-1 times and the sum retrieve times is:

$$RetrieveTimes = \sum_{i=0}^{n-2}(K-i) = \frac{(n-1)(2K-n+2)}{2}$$

Since $n \geq 2$, we have:
Re trieveTimes $\leq K(n-1) < nK$ and the maximal reroute (route) times is n.
When nK is not too much, DLB Algorithm can map a new traffic flow onto an appropriate route quickly.
Also from simulation results, we can see that traffic throughput and network resource utilization is the worst when using Shortest-Path Algorithm. Among our three Load Balancing algorithms, RSLB and DLB are better than TSLB. From two figures 4.4 (c, d), we can see that under heavy load Condition (from 4th second to 6th second), RSLB has the similar performance to DLB. But under light load condition (from 0th second to 4th second), DLB's performance is much better than RSLB, it can fit for traffic flow's burstiness very well.
The main restriction of DLB is that there must not been very large number of parallel routes between Ingress and Egress and each traffic flow must not reroute very frequently, this will result great calculation cost. Fortunately, the number of parallel routes will not be too large in general and traffic will not reroute frequently. So, we conclude that under no priority condition DLB is the best in these three Load Balancing algorithms.

8. SUMMARY OF ANALYSIS

	TSLB	RSLB	DLB
Traffic flow's burstiness	poor	average	Good
Reduces congestion	good	good	Excellent
Utilization of n/w resource.	low	sufficient	good
Traffic Throughput	low	Average	High
Bandwidth request	Low	Average	High
Light load condition performance	Poor	Average	Excellent
Heavy load condition performance	Poor	Excellent	Excellent

Various other algorithms which are proposed by various researchers meant for congestion control for better QOS and also for multicast traffic are advanced form of DLB and PIM-SSM protocol respectively.

9. CONGESTION CONTROL

The proposed algorithm Dynamic Load Balancing Algorithm implements a local search technique where the basic move is the modification of the route for a single Label Switched Path. Experiments under a dynamic traffic scenario show a reduced rejection probability especially with long-lived connection requests, thus providing a better use of resources when compared to existing constraint-based routing schemes for traffic engineering in MPLS networks
The IETF RFC 3272 classifies congestion control schemes according to the following criteria [2]:

a) Response time scale
b) Reactive vs. Preventive
c) Supply side vs. Demand side

The proposed algorithm was as under:

```
1. <congestedLinkSet> ← calculateNetworkLoad (x)
2. bestCandidateLoad ← +∞
3. candRerSet ← ∅
4. for each link (cFrom, cTo) ∈ congestedLinkSet
5.       for each LSP_i crossing (cFrom, cTo)
6.            removePartialLoad (LSP_i)
7.            find an alternate path A_LSP_i for LSP_i
8.            vl ← load on the alternate path A_LSP_i
9.            if (vl = bestCandidateLoad)
10.               candRerSet ← candRerSet ∪ {A_LSP_i, LSP_i}
11.           else if (vl < bestCandidateLoad)
12.               bestCandidateLoad ← vl
13.               candRerSet ← {A_LSP_i, LSP_i}
14.           restorePartialLoad (LSP_i)
15. if (candRerSet ≠ ∅)
16.      {A_LSP_j, LSP_j} ← pickRandomElement (candRerSet)
17.      reroute traffic from LSP_j to A_LSP_j
```

The performance of the proposed algorithm is evaluated through two different set of experiments.

1) The first is focused on the feasibility of DYLBA in a simulated MPLS network context by using an extension of the network simulator called MNS [21, 22]. This simulator has the advantage to reproduce all the signalling mechanisms needed to set-up or tear-down LSPs in an MPLS network.

2) The second set of experiments is based on a simulation program implemented in C++ and used to verify the path set-up rejection ratio of the proposed DYLBA algorithm compared with both Minimum-Hop routing Algorithm (MHA) and Minimum Interference Routing Algorithm (MIRA).

Simulation results show that our algorithm performs better than MIRA in specific condition of network traffic, by reducing the LSP rejection probability and the average end-to-end delays. An interesting direction to extend our work is to consider the use of our algorithm in the context of G-MPLS optical networks, where one can integrate resource information at both the IP and the optical layers.

10.DIFFERENTIATED MULTICAST SERVICES

Protocol-Independent Multicast (PIM) is a family of multicast routing protocols for Internet Protocol (IP) networks that provide one-to-many and many-to-many distribution of data over a LAN, WAN or the Internet. It is termed protocol-independent because PIM does not include its own topology discovery mechanism, but instead uses routing information supplied by other traditional routing protocols such as the Routing Information Protocol, Open Shortest Path First, Border Gateway Protocol and Multicast Source Discovery Protocol.

PIM source-specific multicast (PIM-SSM) builds trees that are rooted in just one source, offering a more secure and scalable model for a limited amount of applications (mostly broadcasting of content). In SSM, an IP datagram is transmitted by a source S to an SSM destination address G, and receivers can receive this datagram by subscribing to channel (S, G).

[23] Presents an algorithm for deploying IP multicast traffic distributed over Diff-Serv aware MPLSTE network. Introduces showing in the same time how the LSPs of the MPLS machinery can be engaged in order to provide guaranteed QoS down to the level of clients. Although the LSP path was configured to have a static behaviour, a next step might be to modify the establishment of these traffic trunks based LSPs to become more dynamic.

11.CONCLUSION

Three Load Balancing algorithms are proposed and evaluated. They enhanced both the network throughput and resource utilization. These three algorithms only fit for no traffic priority condition. The load-balancing algorithms for different priority and multicast condition are needed for further study.

Aimed to the limitations of traditional routing algorithms and to perfect MPLS application in the domain of load balance, an algorithm of static load balance based on topology is designed in this dissertation. When the shortest path is congested in the network, the algorithm is used to find a low load sub shortest path for the congestion path based on the bandwidth occupied situation and the topology. By using MATLAB to simulate the algorithms and compared with traditional algorithm, this algorithm can better ease the network congestion problems because of unbalanced allocation of network resource.

Also to overcome the drawbacks of static load balance based on topology resource static load balanced algorithm was studied which was much better than TSLB. But there are still many features that are not covered by the Static load balanced algorithms due to which Dynamic load balanced algorithm is also studied and all three algorithms are simulated using NS2 and DLB proves to be the best one but also with the condition of no priority.

With priority condition like congestion control and differentiated multi cast services various other algorithms are taken into consideration whose simulation are been either NS2 or Matlab.

REFERENCES

[1] D. Awduche *et al.*, "Overview and Principles of Internet Traffic Engineering," IETF RFC 3272, May 2002.

[2] Y. Lee *et al.*, "Traffic Engineering in Next-Generation Optical Networks," *IEEE Commun. Surveys & Tutorials*, vol. 6, no. 3, 2004, pp. 16–33.

[3] Y. Wang and Z. Wang, "Explicit routing algorithms for Internet traffic engineering," *IEEE International Conference on Computer Communications and Networks (ICCCN)*, 1999.

[4] Y. Wang and M. R. Ito, "Dynamics of load sensitive adaptive routing," *IEEE International Conference on Communications (ICC)*, 2005.

[5] B. Fortz and M. Thorup, "Optimizing OSPF/IS-IS weights in a changing world," *IEEE J. Sel. Areas Commun.*, vol. 20, no. 4, pp. 756-767, 2002.

[6] J. Chu and C. Lea, "Optimal link weights for maximizing QoS traffic," in *Proc. IEEE ICC 2007*, pp. 610-615, 2007.

[7] A. K. Mishra and A. Sahoo, "S-OSPF: a traffic engineering solution for OSPF based on best effort networks," in *Proc. IEEE Globecom 2007*, pp. 1845-1849, 2007.

[8] M. Antic and A. Smiljanic, "Oblivious routing scheme using load balancing over shortest paths," *IEEE ICC 2008*.

[9] M. Antic and A. Smiljanic, "Routing with load balancing: increasing the guaranteed node traffics," *IEEE Commun. Lett.*, vol. 13, no. 6, pp. 450-452, June 2009.

[10] D. Awduche, L. Berger, T. Li, V. Srinivasan, and G. Swallow, "RSVPTE: Extensions to RSVP for LSP Tunnels," RFC 3902, Dec. 2001.

[11]Keping Long, Zhongshan Zhang and Shiduan Cheng, "Load balancing algorithms in MPLS traffic engineering," In 2001 IEEE Workshop on High Performance Switching and Routing, 2001, pp.175-179.

[12] Aboul-Magd O. et al., Constraint-Based LSP Setup using LDP. Internet Draft, working in progress, June 2000.

[13] Introduction to Multi Protocol Label Switching by Tripti Batra and Gagan Aggarwal in Sep 2006.

[14]D. Awduche et al., Requirements for Traffic Engineering Over MPLS. IE TF RFC 2702, Sept. 1999.

[15] T. Li and Y. Rekhter, A Provider Architecture for Differentiated Serviced and Traffic Engineering (PASTE). IETF RFC 2430, Oct. 1998.

[16] Xipeng Xiao, Alan Hannan. Traffic Engineering with MPLS in the Internet, IEEE Network, March/April 2000.

[17] R.Callon et al., A Framework for Multiprotocol Label Switching.Internet Draft, working in progress, Sept. 1999.

[18] E. Rosen, A.Viswanathan and R. Callon. Multiprotocol Label Switching Architecture. Internet Draft, working in Progress, August 1999.

[19] UCB, LBNL, VINnT Network Simulator-ns2. http://wwwmash. cs.berkeley.edu/ns/ns.html 179.

[20] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. Overview and Principles of Internet Traffic Engineering. IETF RFC 3272, May 2002.

[21] G. Ahn and W. Chun. Design and Implementation of MPLS Networks Simulator (MNS). *http://flower.ce.cnu.ac.kr/~fog1/mns/*.

[22] V. Project. Network Simulator - V.2.1b9. *http://www.isi.edu/nsnam/ns/*.