Global QoS model in the ISP networks: DiffServ aware MPLS Traffic Engineering

Kyeongja Lee

ECOLE CENTRALE DE LILLE
UNIVERSITE DES SCIENCES ET TECHNOLOGIES DE LILLE

N° d'ordre 0025 Année : 2006

THESE

Pour obtenir le grade de

DOCTEUR DE L'ECOLE CENTRALE DE LILLE

Doctorat délivré conjointement par l'Université des Sciences et Technologies de Lille et l'Ecole Centrale de Lille

Discipline : *Automatique et Informatique industrielle*

présentée et soutenue le 2 novembre 2006

par

KyeongJa LEE

Modèle global pour la Qualité de Service dans les réseaux de FAI : intégration de DiffServ et de l'ingénierie de trafic basée sur MPLS

devant le jury composé de :

Mr.	Thierry DIVOUX	*Rapporteur*	*Professeur à l'Université Henri Poincaré*
Mr.	Guy JUANOLE	*Rapporteur*	*Professeur à l'Université Paul Sabatier*
Mr.	Olivier BONAVENTURE	*Examinateur*	*Professeur à l'Université Catholique de Louvain*
Mr.	Pascal YIM	*Examinateur*	*Professeur à l'Ecole Centrale de Lille*
Mr.	Armand TOGUYENI	*Directeur*	*Professeur à l'Ecole Centrale de Lille*
Mr.	Ahmed RAHMANI	*Co-directeur*	*Maître de conférences HdR, à l'Ecole Centrale de Lille*

Directeurs de thèse : Armand TOGUYENI et Ahmed RAHMANI

Thèse préparée au Laboratoire d'Automatique, Génie Informatique et Signal

LAGIS, UMR 8146 – Ecole Centrale de Lille

Contents

List of Figures

List of Tables

Glossaries

AF	Assured Forwarding
AIMD	Additive Increase Multiplicative Decrease
BC	Bandwidth Constraints
BGP	Border Gateway Protocol
CBR	Constraint based routing
DiffServ	Differentiated Service
DSCP	DiffServ Code Point
DS-TE	DiffServ aware MPLS Traffic Engineering
EF	Expedited Forwarding
FEC	Forwarding Equivalence Class
IETF	Internet Engineering Task Force
IntServ	Integrated Service
ISP	Internet Service Provider
LBWDP	Load Balancing over Widest Disjoints Paths algorithm
LDM	Load Distribution in MPLS network
LDP	Load Distribution Protocol
LP	Linear Programming
LSP	Label Switched Path
MATE	MPLS Adaptive Traffic Engineering
MNS	MPLS Network Simulator
MPLS	Multi-Protocol Label Switching
MPLS TE	Traffic Engineering based on MPLS
NS2	Network Simulator version 2
OSPF	Open Shortest Path First
PEMS	PEriodic Multi-Step algorithm for DS-TE network
PER	Prediction of Effective Repartition
PHB	Per-Hop Behaviour
QoS	Quality of Service
RED	Random Early Detection

RFC	Request For Comments
RSVP	ReSource reserVation Protocol
SLA	Service Level Agreement
SPN	Stochastic Petri Nets
SWP	Shortest Widest Path
TB	Traffic Bifurcation
TCP	Transmission Control Protocol
TE	Traffic Engineering
TEWG	Traffic Engineering Working Group
WDP	Widest Disjoint Path
WSP	Widest Shortest Path

General Introduction

"Best-Effort" service of today's Internet can provide no guarantees regarding loss rate, bandwidth, delay, delay jitter, and so on. Today's various multimedia applications such as digital video and audio expect to be supported multiple and diverse Quality of Service (QoS) requirements better than "best-effort" service. These applications need to find appropriate and available network resources and reserve them in order to support and guarantee their specific services.

Traffic Engineering (TE) is concerned with the performance optimization of operational networks. In this study we assume that routing paths can be setup to transport the traffic from a source to a destination. The purpose of TE is to increase service availability and to reduce the load of each router. Changing path to lower cost [Salama, 1997] [Vutukury, 1999] or sending traffic along multiple paths [Rao, 1998], setting cost according to bandwidth availability [Fortz, 2000] are various manners to realize traffic engineering. We focus on multipath routing. When disseminating traffic into multiple paths, routers should adaptively allocate flows to each path in order to achieve load balancing among these paths.

This study concerns IP routing in an ISP (Internet Service Provider) network. Schematically this type of network can be considered as composed of several pairs of ingress-egress routers and core routers that enable to transport the traffic between the pairs of peripheral routers. To achieve this function, let us assume that this requires some kinds of connections in the connectionless IP networks. In particular, MPLS (Multi-Protocol Label Switching) supports efficiently, the explicit route setup between ingress and egress routers, and hence the ability to balance traffic through the network.

The general schematic approach of multipath routing algorithm is proposed that it has two steps: computation of multiple candidate paths and traffic splitting among these multiple paths [Lee, 2005a]. A multipath routing algorithm selects candidate paths based on some metrics and then distributes the traffic demand over the selected paths. Routing control in TE essentially balances traffic load over multiple paths in the network to minimize its congestion. It concerns how to select good paths and to distribute traffic among those paths such that given quality of service (QoS hereafter) constraints are met or close to the target at a large possible extent while at the same time optimizing some global, network-wide objectives such as utilization [Peerapon, 2002]. In this study, the objective of traffic engineering is to minimize the utilization of the most heavily used link in the network. Intuitively the hot spots are the points with the highest link utilization. Reducing the link utilization at these points balances the traffic distribution across the network.

The various algorithms in the literature give solutions for effectively computing the multipaths and ways to minimize delay and increase throughput. The problem formulation of traffic bifurcation is identified and some existing multipath algorithms are compared in order to verify if they are scalable and efficient and extended use.

A new routing paradigm that emphasizes searching for acceptable paths satisfying various QoS requirements is needed for integrated communication networks. Given the QoS request of a flow or an aggregation of flows, paths that will provide best service to the flows must be selected. Each multipath routing algorithm in the literature is declared very efficient by its authors but generally with respect to restricted conditions. In this context, instead of developing a new routing algorithm, we propose a hybrid approach combining the best features of some algorithms to construct more performing hybrid algorithms. Indeed some algorithms could be more efficient than original one in the point of traffic engineering. Three hybrid algorithms are proposed as efficient multipath routing algorithms and simulated to estimate their performances.

MPLS offers many advantages to service providers. However, it is incapable of providing differentiated service levels in a single flow. Hence MPLS and DiffServ seem to be a perfect match and if they can be combined in such a way to utilize each

technology's strong points and counter the other's weaknesses, it can lead to a goal of end to end QoS feasible. DiffServ-aware MPLS Traffic Engineering is needed to differentiate the quality of service according to today's various requirements.

If the given metric is a dynamic metric that changes according to network status, accurate value is essential to select the path. Through this dissertation, when selecting the candidate paths, we focus on their disjointedness to avoid the influence of link failure, and when distributing the traffic we focus on prediction power to avoid the influence of inaccurate state information.

We propose new DS-TE model, PEMS (PEriodic Multi-Steps routing for DS-TE network), to verify that DiffServ aware MPLS traffic engineering is useful to meet the customers' various requirements.

This study focuses on efficient QoS routing mechanism. The interest is in the problems of multiple constraints and inaccurate network information. The research goal of this dissertation is as follows.

1. establish the general multipath routing scheme,
2. propose the hybrid approach algorithm by combination of cross steps from different routing algorithms,
3. propose an efficient routing solution for DiffServ aware MPLS Traffic Engineering network for differentiating the QoS according to the class.

In summary, this dissertation not only discusses on some major issues related to QoS, but also presents a systematic approach for providing QoS in the Internet.

The remainder of the dissertation is organized as follows. Chapter 1 presents the terminology and basic concept of routing problem and need of QoS and traffic engineering. The schematic workflow of multipath routing is presented and in this scheme some existing multipath routing algorithms are compared in Chapter 2. New multipath routing algorithms based on combination of each step of different solutions are proposed and new traffic splitting modular algorithm, called PER (Prediction of Effective Repartition), to be used as one modular of hybrid algorithm is proposed. This PER algorithm is used also in PEMS algorithm in chapter 4. In Chapter 4, the

integration of MPLS and DiffServ is studied. And new DS-TE routing algorithm, PEMS, is proposed. It includes new link disjointed path retrieve algorithm to select some good paths to be searched online. PEMS is compared with LBWDP (Load Balancing over Widest Disjoints Paths algorithm), one of the hybrid algorithm proposed in chapter 3, which does not support the service differentiation through the simulation. The performance of the algorithms proposed in this study is evaluated through simulation using *ns-2* and *MNS2.0* simulators. Chapter 5 concludes this dissertation.

Chapter 1:

Quality Of Service in the Internet

1.1. Introduction

OSPF (Open Shortest Path First) [RFC2328] and other dynamic routing protocols in the current Internet always forward packets to the shortest path. Shortest path routing leads to imbalanced traffic distribution. This can cause congestion in some parts of the network even if traffic load is not particularly heavy. This can cause problems for flows with a need for QoS (Quality of Service) guarantees if the shortest path does not have enough resources to meet the requirements. Companies that do business on the Web need QoS for better delivery of their content and/or services to attract more customers. And ISPs (Internet Service Providers) are facing the challenge of offering improved quality of service to their customers and they need the value-added services in their networks to increase revenue through supporting QoS.

On the Internet and in other networks, QoS is the idea that transmission rates, error rates, and other characteristics can be measured, improved, and, to some extent, guaranteed in advance. QoS is of particular concern for the continuous transmission of high-bandwidth video and multimedia traffic demands. Transmitting this kind of content dependably is difficult in public IP networks using ordinary "best effort" protocols.

The IETF (Internet Engineering Task Force) has proposed many service models and mechanisms to meet the demand for QoS. The architectures and mechanisms developed for enabling QoS have two key QoS issues: resource allocation and performance optimization. For resource allocation in the internet, Integrated Service (IntServ) [RFC1633] and Differentiated Service (DiffServ) [RFC2475] are proposed by IETF.

MPLS (Multi-Protocol Label Switching) and Traffic Engineering give ISPs a set of management tools for bandwidth provisioning and performance optimization. They can support QoS on a large scale and at reasonable cost.

The essence of IntServ is to reserve resources for each individual flow so that the service quality can be guaranteed if needed. The essence of DiffServ is to divide traffic into different classes and give them differentiated treatments [RFC3031].

1.2. Quality of Service (QoS)

The growth of multimedia applications over wide area networks has increased research interest in QoS. Just increasing the amount of resources such as available bandwidth to avoid congestion does not provide proper resource utilization. This over-provisioning solution has been effective in applications such as FTP (File Transfer Protocol), HTTP (Hyper-Text Transfer Protocol), and e-mail. However, the characteristics of traffic over the Internet have changed. There are many new types of applications, each with very different operational requirements. The Internet has become the backbone of future communications on ubiquitous environment. Mobile, ad-hoc networks and hand-held devices such as PDA have different QoS requirements. Multimedia applications require network services beyond what IP delivers.

The communication delay and synchronization needed for voice, data and images are major concerns. Internet telephony (Voice over IP) and other multimedia applications such as video conferencing, video-on-demand and media streaming require service guarantees and have strict timing requirements. The size and quality of display devices, and resources such as CPU (Central Processing Unit), battery power and bandwidth are always limited.

Traffic	Application	Requirement
Voice	IP Telephony	Low delay, Low jitter (real-time)
Video	Video on Demand Video Conferencing	Low delay Guaranteed bandwidth

Traffic	Application	Requirement
Data	Web Access, e-mail File Transfer Data Backup	Low packet loss Availability

Table1.1 Special requirements for the applications

QoS can be parameterized as throughput, delay, delay variation (jitter), loss and error rates, security guarantees and so on, that are acceptable in an application. As such, QoS depends on characteristics of applications. For instance, the variation in delay, the difference between the largest and the smallest delay, is called delay jitter and jitter is an important quality for IP (Internet Protocol) telephony, which can tolerate a certain percentage of packet loss without any degradation of quality. For data transfer, loss is a crucial QoS parameter. QoS control requires an understanding of the quantitative parameters at the application, system and network layers.

1.2.1. IntServ (Integrated Service)

Integrated Service develops a new architecture for resource allocation to meet the requirements of real-time applications. Real-time applications refer to a group of applications that have stringent delay requirements. Typically they have a deadline for data to arrive by, after which the data become less useful. An example of real-time applications is audio and video conferencing. For a typical audio-streaming application, the source makes the voice signal to the packets and sends the packets to the network. Since individual packets experience a different delay amount in the network, the original gap between packets is no longer preserved when the packets get to the destination. Because of the distortion of timing caused by the delay jitter, the quality of the voice signal would not be good if the receiver simply sent the data to the audio device as the packets came in. The goal of IntServ is to preserve the datagram model of IP-based networks and at the same time support resource reservation for real-time applications.

1.2.1.1. Framework

The fundamental idea of IntServ architecture is to reserve resources such as bandwidth and buffers. To receive performance assurance from the network, an application must set up the resources reservation along its path before it can start to transmit packets. It is based on the use of the ReSource reserVation Protocol (RSVP) [Tanenbaum, 1996] which is invented as a signaling protocol for applications to reserve resources. The application initiates a session on demand with the network using RSVP. This session identifies the service requirements of the application including the bandwidth and delay. RSVP is a receiver-oriented reservation protocol for setting up resource reservation in the Internet, that is, the receivers are responsible for deciding what resources will be reserved. The acknowledgements from receivers travel from the receivers toward the sender and gradually build up a path.

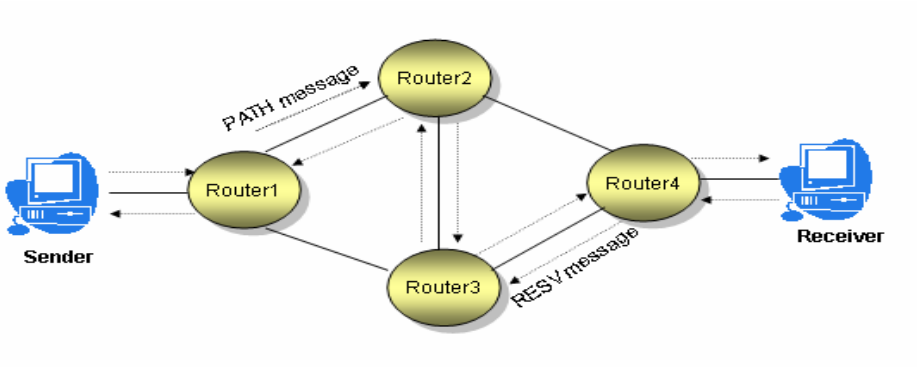


Fig.1.1 Basic Resource Reservation Setup Operations of RSVP signaling protocol

The sender sends a PATH message to the receiver specifying the characteristics of the traffic. Every intermediate router along the path forwards the PATH message to the next hop determined by the routing protocol. Upon receiving a PATH message, the receiver responds with a RESV message to request resources for the flow. Every intermediate router along the path can reject or accept the request of the RESV message. If the request is rejected, the router will send an error message to the receiver, and the signaling process will terminate. If the request is accepted, link bandwidth and buffer space are allocated for the flow and the related flow state information is installed in the router.

1.2.1.2. Service Classes [RFC1633]

IntServ provides two service classes in addition to best-effort service, that are the Guaranteed Service, for applications requiring fixed delay bound, and Controlled Load service, for applications requiring a reliable and enhanced best-effort service.

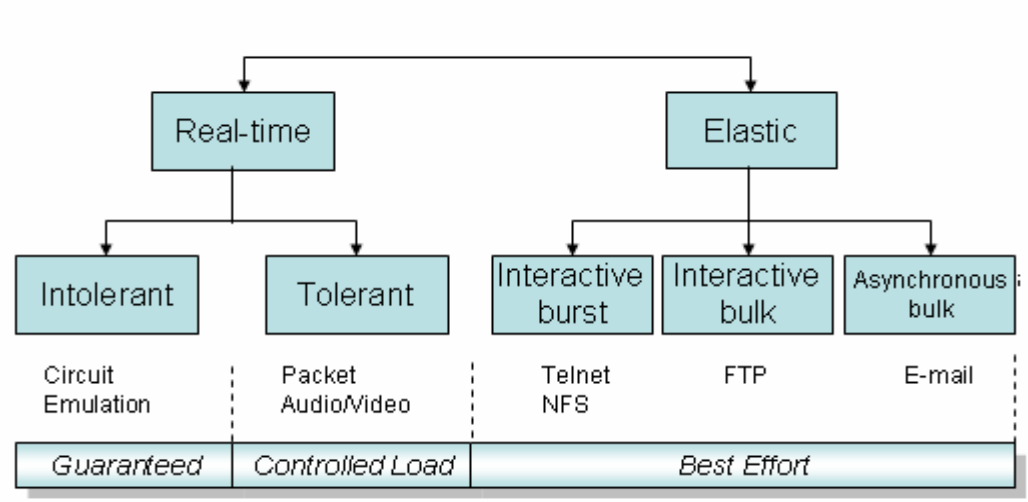


Fig.1.2 Traffic sources for IntServ

The Guaranteed Service is defined to provide an assured level of bandwidth, a firm end-to-end delay bound and no queuing loss. It is intended for real-time applications such as voice and video.

The Controlled Load Service definition does not include any firm quantitative guarantees but rather “the appearance of a lightly loaded network”. It intends to support a broad class of applications that were originally developed for today’s Internet, but is highly sensitive to network overload. It was intended for applications that could tolerate a limited amount of loss and delay, including adaptive real-time applications. It strives to approximate tightly the behavior visible to applications receiving the Best-Effort service during unloaded conditions.

1.2.1.3. Problems

The IntServ architecture has satisfied both necessary conditions for the network QoS, i.e., it provided the appropriate bandwidth and queuing resources for each application

flow. However, the IntServ implementations with RSVP required the per-microflow state and signaling at every router. The problems of IntServ are as follows.

- The amount of state information increases proportionally with the number of flows. It needs a huge storage and processing load on the routers. This causes significant complexity to network operation and it can be the problem of unscalability.
- All routers must have RSVP, admission control, packet classification and packet scheduling. The requirements on every router are high.

Therefore, the IntServ model was implemented only in a limited number of networks, and naturally the IETF moved to develop DiffServ as an alternative QoS approach with minimal complexity.

1.2.2. DiffServ (Differentiated Service)

The main goal of DiffServ is to provide a scalable framework for offering a range of services in the Internet with Quality of Service support and without the need to maintain per-flow state in every router. DiffServ has only a limited number of service classes indicated by the DS field. Since resources are allocated in the granularity of class, the amount of state information is proportional to the number of classes rather than the number of flows. So DiffServ is more scalable than IntServ.

The DiffServ model is based on redefining the meaning of 8-bit TOS (Type Of Service) field in the IP header. The original TOS definition was not widely implemented, and now the field is split into the 6-bit DiffServ Code Point (DSCP) value and 2-bit unused part.

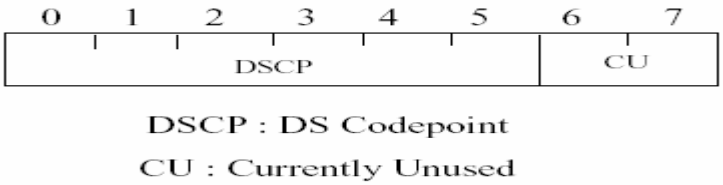


Fig.1.3 DS field [RFC2474]

In a DiffServ network, the boundary nodes (or edge nodes) and interior nodes (or core nodes) have different tasks. DiffServ achieves scalability through performing

complex QoS functions such as classification, marking, and conditioning operations using the DiffServ Code Point (DSCP) into a limited number of traffic aggregates or classes only at the edge nodes. In the core routers, scheduling and queuing control mechanisms are applied to the traffic classes based on the DS field marking: all traffic conditioning and dropping is intelligently handled at the network layer using IP DiffServ QoS mechanisms.

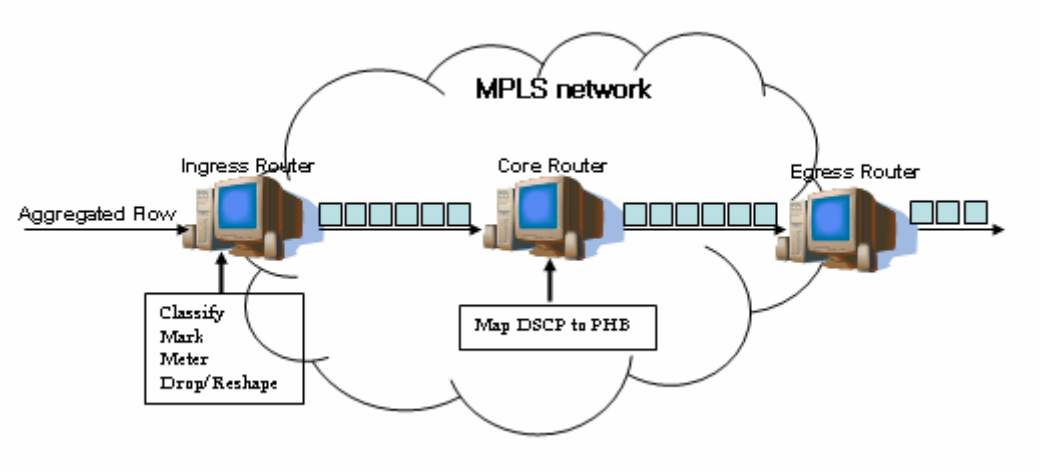


Fig.1.4 Edge and core nodes of DiffServ

DiffServ's ingress nodes process and mark the TOS byte in IP packet header by a DSCP, based on a negotiated contract. Other routers in the domain that receive the packet are only concerned with the DSCP value to assign a particular treatment to the packet. This particular treatment is called a Per-Hop-Behavior (PHB).

[Abdellatif, 2002] presents the definition of a Stochastic Petri Nets (SPN) based framework for QoS analysis in packet switched networks. This is central to the design of networks with such capabilities; traffic shaping, traffic policing, packet scheduling, buffer management, admission control and routing. In [Abdellatif, 2005], authors extend this SPN model to assure DiffServ functions.

1.2.2.1. Traffic Conditioning

In ISP (Internet Service Provider) domain, ingress routers are responsible for mapping packets to one of the forwarding classes supported in the network, and they must ensure that the traffic conforms to the SLA (Service Level Agreement) for the specific customer. Once the packets pass the boundary nodes into the interior of the

network, resource allocation is performed based only on forwarding classes. These functions performed by boundary nodes are referred as traffic classification and traffic conditioning.

In Fig.1.5, a *Classifier* classifies the incoming packets into an appropriate behavioral aggregate and identifies the PHB treatment to be assigned to the flow.

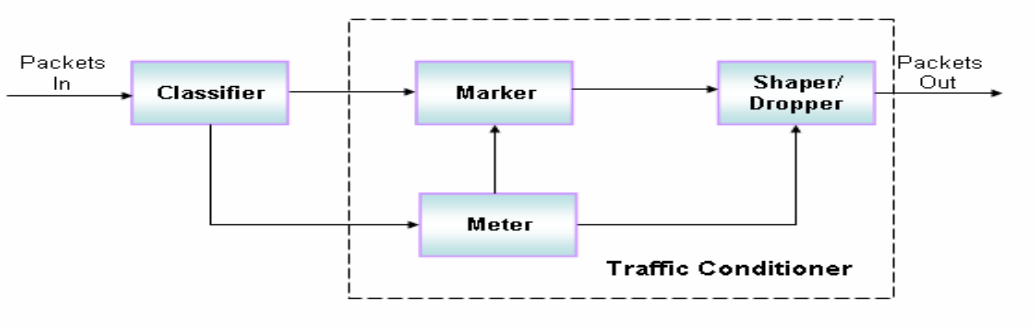


Fig.1.5 Logical View of a Packet Classifier and Traffic Conditioner [RFC2475]

A Traffic Conditioner may condition the incoming flow to ensure the traffic meets the profile agreed upon in the SLA or equivalent agreement.

A *Traffic Conditioner* consists of four basic elements: meter, marker, shaper and dropper.

- Meter: The meter monitors and measures each traffic stream selected by a classifier against a traffic profile and decides that the packet is in profile or out of profile. Then it informs the state information to other traffic conditioning elements. Out of profile packets may be dropped, remarked for a different service, or held in a shaper temporarily until they become in profile. In profile packets are put in different service queues for further processing.

- Marker: The marker sets the DS field of a packet to a particular code-point, then adds the marked packet to a forwarding class. Markers may act on unmarked packets or remark previously marked packet. Marking may occur at many different locations. Marking is also one of the actions that can be taken on nonconforming packets. When a traffic stream passes a

meter, some packets may be marked with a special DSCP to indicate their nonconformance.

- Shaper: The shaper is used to smooth the traffic stream at a particular aggregate level. A shaper is to delay some or all of packets in a packet stream in order to bring the stream into compliance with its traffic profile. It usually has a finite buffer, and packets may be discarded if there is insufficient buffer space to hold the delayed packets.

- Dropper: The dropper makes discarding decisions based on the content of the service level and SLAs (Service Level Agreements). A dropper can be implemented as a particular case of a shaper by setting shaper buffer size to zero packets. It just drops out-of-profile packets. The function of a dropper is known as traffic policing.

1.2.2.2. PHB (Per Hop Behavior)

DiffServ only defines DS fields and PHBs. It is ISP's responsibility to decide what services to provide. Using the classification, policing, shaping and scheduling mechanisms, many services can be provided. DiffServ has three kinds of services.

- *EF (Expedited Forwarding) : premium* service with reliable, low delay and low jitter delivery,
- *AF (Assured Forwarding) : assured* service with reliable and timely delivery,
- *Default (Best-effort)*: it is the normal service offered by IP networks.

Because there are only 3 classes of traffic, only the first 3 bits of the DSCP are used to denote the traffic class that each packet belongs to and the drop preference of the packet. The other 3 bits of the DSCP are set to 0. In other words, only 8 class selector code points are used. For each output port, one queue is used for each class. Of the 3 bits used, the first two bits are used to denote traffic classes (and thus for selecting the queue associated with those classes), the third bit is used to indicate the drop preference inside each class. Policing is to determine whether traffic from a particular customer

conforms to the traffic profile specified in the SLA, and take action accordingly. Marking is to set the DSCPs of the packets.

Token bucket represents the policing function of Traffic Conditioning Block of DiffServ. A token bucket flow is defined by (*r*, *b*), where *r* denotes the rate at which tokens are accumulated and *b* is the depth of the token pool (in bytes).

For premium service, policing should be implemented with a leaky bucket so that burst will not be introduced, or with a token bucket that allows only small burst. When a packet arrives and there are tokens in the bucket, the packet is considered as conformant. The DSCP of the packet is set to 111000. If a packet arrives and there is no token in the bucket, then the packet is considered as nonconformity. Depending on the SLA between the ISP and the customer, nonconforming packets may be dropped immediately or may be transmitted and accounted for extra charge. If nonconforming packets are to be transmitted, their DSCPs will also be set to 111000. All packets are put into a queue called premium queue (PQ).

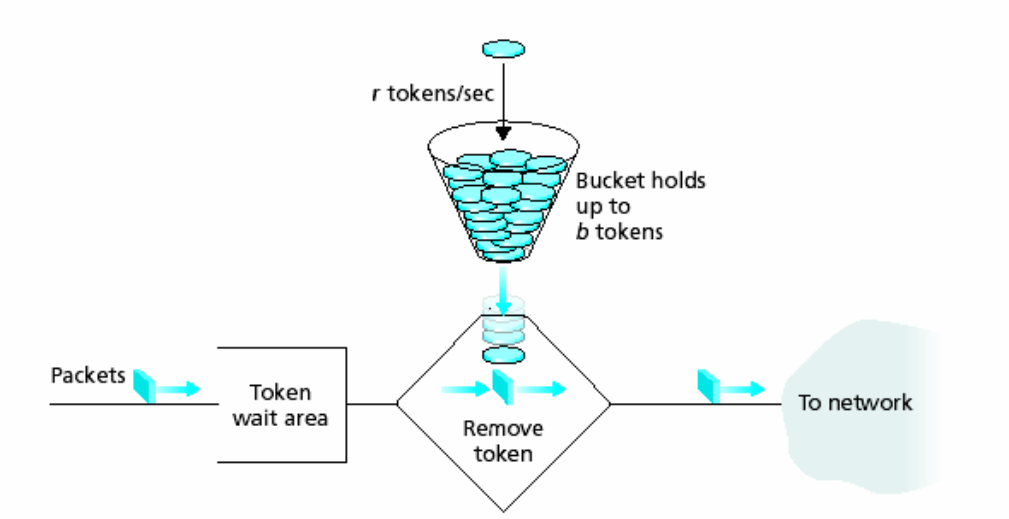


Fig.1.6. Leaky bucket policer

For assured service, policing should be implemented with a token bucket so that some kind of burst is allowed. When a packet arrives and there are tokens in the bucket, the packet is considered as in profile. The DSCP of the packet is set to 101000 by the marking process. If a packet arrives and there is no token in the bucket, then the packet is considered as out-of-profile. The marking process will set the DSCP of the packet to

100000. Note that the first two bits are the same for both in-profile packets and out-of-profile packets, because both types of packets belong to the same traffic class.

But the third bit is different, so that in-profile and out-of-profile packets will have different drop preference. The Random Early Detection (RED) algorithm then determines whether to drop a particular packet or to queue it. No packets with DSCP 101000 will be dropped before all 100000 packets are dropped. All the packets that are not dropped no matter they are in or out of profile, are put into the same queue to avoid out of order delivery. This queue is called assured queue (AQ).

Best effort traffic may also be policed depending on the SLAs. In-profile packets will have DSCP 001000, and out-of-profile packets will have DSCP 000000. Alternatively, no policing is done for best effort traffic and all packets have their DSCPs set to 000000. RED is applied, and packets that are not dropped enter a queue called default queue (DQ).

Because the PQ, AQ and DQ all have fixed output rate, premium, assured and best effort traffics are all shaped.

1.2.2.3. Comparison with IntServ

The approach IntServ and DiffServ are quite different from each other.

- DiffServ is allocated in the granularity of a class and the amount of state information is proportional to the number of classes rather than the number of flows. DiffServ hence is more scalable than IntServ and can handle large number of flows. Also, since PHBs are essentially kept single, DiffServ lends itself well to use at high speeds making it scalable in terms of speed. So DiffServ is the preferred technology for large-scale IP QoS deployments today, such as service provider backbone.
- In DiffServ, only boundary nodes classify traffic and mark packets. Once the packets are marked, the core nodes use the forwarding classes encoded in the packet header to determine the treatment of the packets. IntServ requires all nodes to perform packet classification to identify packets from reserved flows and schedule them with per-flow queuing.

- DiffServ provides resource assurance through provisioning combined with prioritization rather than per-flow reservation as in IntServ. By allocating resources to forwarding classes and controlling the amount of traffic for these classes, DiffServ creates different levels of services and resource assurance but not absolute bandwidth guarantees or delay bounds for individual flows.

1.2.2.4. Limitation of DiffServ

DiffServ is a scalable solution that does not require per-flow signaling and state maintenance in the core. But it appears to be principal limitation to the DiffServ architecture.

DiffServ architecture suggests only mechanisms for relative packet forwarding treatment to aggregate flows, traffic management and conditioning. However it does not provide architecture for end-to-end QoS.

DiffServ framework does not lend itself to handle link failures. For example if a link carrying *premium service* traffic, in DiffServ domain goes down, there is no way for the provider to quickly send the traffic through alternate link and ensure minimum packet loss.

Furthermore, there is no traffic engineering provision in DiffServ. As a result some links in the domain might experience congestion while other links go unutilized.

While QoS schemes try to provide better service under congestion for flows with QoS requirements, it would be even more desirable to avoid these situations all together.

1.3. Traffic Engineering

Traffic Engineering (TE) is a solution to install and check the flows of traffic inside the network, taking full advantage of all the potentialities offered by the available resources and avoiding uneven network utilization. TE is needed in the Internet mainly because current IGPs (Interior Gateway Protocols) always use the shortest paths to forward traffic.

Specifically, TE provides the ability to move traffic flows away from the shortest path selected by the IGP and onto a potentially less congested physical path across the

service provider's network. TE is a powerful tool that can be used by ISPs to balance the traffic load on the various links, routers, and switches in the network so that none of these components is over-utilized or under-utilized. In this way, ISPs can exploit the economies of the bandwidth that has been provisioned across the entire network. TE should be viewed as assistance to the routing infrastructure that provides additional information used in forwarding traffic along alternate paths across the network.

This is a definition of TE employed by the Traffic Engineering Working Group (TEWG) within the IETF [RFC2205].

> *Internet traffic engineering is concerned with the performance optimization of operational networks. It encompasses the measurement, modeling, characterization, and control of Internet traffic, and the application of techniques to achieve specific performance objectives, including the reliable and expeditious movement of traffic through the network, the efficient utilization of network resources, and the planning of network capacity.*

By performing TE in their networks, ISPs can greatly optimize resource utilization and network performance. The common optimization objectives include:

- Minimizing congestion and packet losses in the network,
- Improving link utilization,
- Minimizing the total delay experienced by packets,
- Increasing the number of customers with the current assets.

Congestion is one of the most difficult problems that service providers face in their networks. Two main factors have caused network congestion: inadequate network resources and unbalanced traffic distribution. Although the problem of inadequate network resources must be solved with new capacity or by reducing and controlling the demands, unbalanced traffic distribution can be addressed through better management of the resources in the network. With carefully arranged traffic trunks, service providers can spread traffic across the network to avoid hot spots in parts of the network.

The main objective of TE is efficient mapping of traffic demands onto the network topology to maximize resource utilization while meeting QoS constraints such as delay, delay variation, packet loss and throughput.

In order to do TE effective, the IETF introduces MPLS, Constraint-based Routing and an enhanced link state IGP.

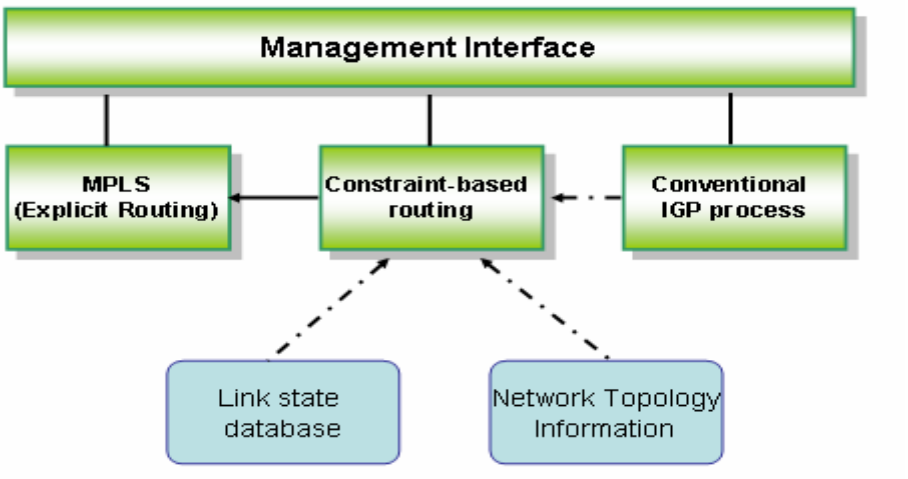


Fig.1.7 Block diagram of Traffic Engineering [RFC2702]

1.3.1. Traffic Engineering in MPLS network

MPLS stands for "Multi-Protocol Label Switching" [RFC3031], and it is applicable to all the protocols of the Network Layer. It is a switching technology aimed at greatly improving the packet forwarding performance of the backbone routers in the Internet or other large networks. The basic idea is to forward the packets based on a short, fixed length identifier termed as a “label”, instead of the network-layer address with variable length match. The labels are assigned to the packets at the ingress node of an MPLS domain. Inside the MPLS domain, the labels attached to packets are used to make forwarding decisions. Thus, MPLS uses indexing instead of a longest address match as in conventional IP routing. The labels are finally popped out from the packets when they leave the MPLS domain at the egress nodes. By doing this, the efficiency of packet

forwarding is greatly improved. Routers which support MPLS are known as "Label Switching Routers", or "LSRs".

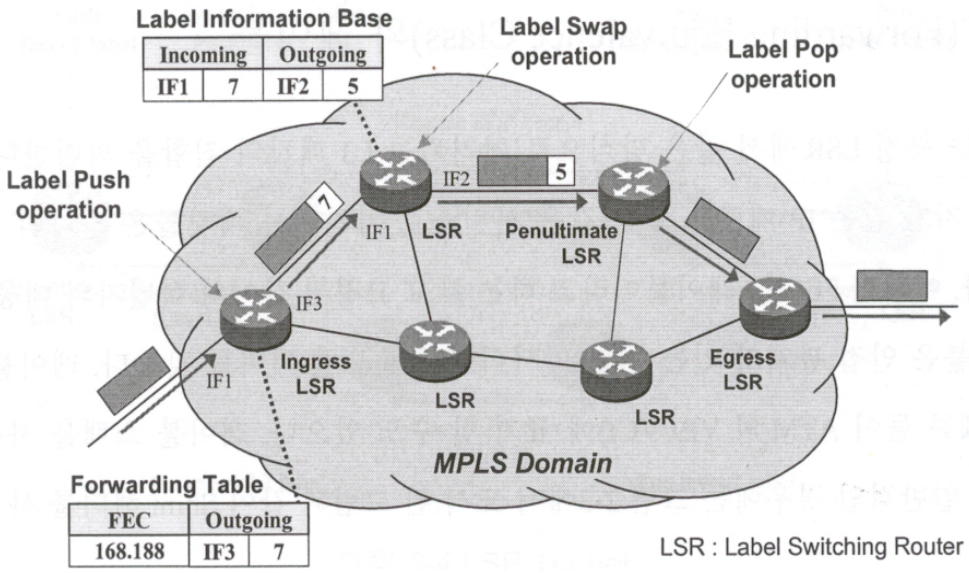


Fig.1.8 MPLS Label switching

The assignment of labels to packets is based on the concept of forwarding equivalence class (FEC). According to this concept, packets which belong to the same FEC are assigned the same label at an ingress node to an MPLS domain. A FEC consists of packets entering a network through the same ingress node and exiting the network through the same egress node. A FEC also consists of packets requiring similar QoS or packet treatment across the MPLS domain. The path traversed by a FEC is called a Label Switched Path (LSP). A signal protocol such as LDP (Load Distribution Protocol) [RFC3036] or RSVP (Resource reSerVation Protocol) [RFC2205] is used to establish and release LSPs.

The Label switching approach was initially conceived in order to improve router performance, but this motivation has diminished with advances in router design and achievement of line-speed forwarding of native IP packets. But later the most important advantage of the MPLS architecture over the native IP forwarding has become apparent: the connection-oriented nature of MPLS allows ISPs to implement TE in their networks

and achieve a variety of goals, including bandwidth assurance, different routing, load balancing, path redundancy, and other services that lead to QoS [Fineberg, 2003].

The main architectural concept underlying MPLS is the clear separation of the control plane from the forwarding plane in network switching elements.

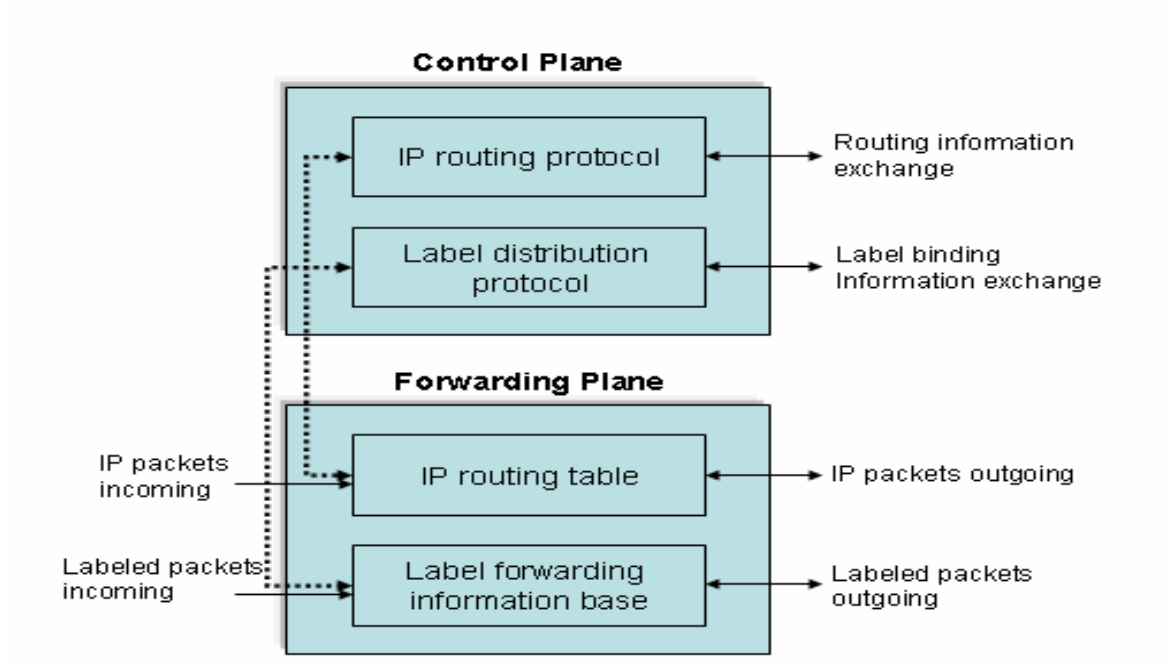


Fig.1.9 MPLS node architecture

The control plane is concerned with network level coordination functions, such as routing and signaling, to facilitate the movement of traffic across the entire network. One of the main functions performed by the MPLS control plane is to facilitate the establishment of label switched paths in MPLS networks. The establishment of LSPs may be subject to various types of preferences and constraints. This means that the control plane needs to distribute and to manage network topology and resource availability information using a routing protocol, and to perform signaling functions to establish and tear-down LSPs. In practice, signaling is one of the most fundamental aspects of the MPLS control plane.

The forwarding components in forwarding plane perform simple label switching operations according to the classical label swapping paradigm.

Load balancing for TE in IP network requires an ability to control traffic flow precisely. In the traditional metric-based control, an administrator can change only link metrics, and changes of some link metrics may affect the overall traffic flow. To manage the performance of a network, it is necessary to have explicit control over the paths that traffic flows traverse so that traffic flows can be arranged to maximize resource commitments and utilization of the network. MPLS has a mechanism called explicit routing that is ideal for this purpose. MPLS uses the label switching approach to set up virtual circuits in IP-based networks. These virtual circuits can follow destination-based IP routing, but the explicit routing mechanism in MPLS also allows us to specify hop by hop the entire path of these virtual circuits. Routing protocols such as OSPF (Open Shortest Path First) [RFC2328] and BGP (Border Gateway Protocol) [RFC1105] determine these explicit routes in advance, and then build tables in each router that define the routes. Packets carry labels to indicate the explicit route they should be taking. Thus, labeled packets follow LSPs.

This procedure of using standard routing protocols to define explicit paths is really the default procedure, and it can take place without operator intervention. In addition, MPLS is flexible enough to define paths based on various constraints such as available bandwidth, the priority setting of packets, the aims of an operator, or the directives of a policy-based server. Thus, MPLS also supports CBR (Constraint-Based Routing).

1.3.2. Constraint-based routing (CBR)

CBR is a routing scheme that considers the QoS requirements of a flow when making routing decisions. As opposed to traditional shortest path routing which only considers the hop-count, CBR is designed to find feasible paths that satisfy multiple constraints, for example bandwidth, QoS requirements such as delay and jitter, or other policy requirements on the paths. While determining a route, CBR considers not only network topology but also requirements of the flow, resource availability of the links, and possibly other policies specified by the network administrators. Therefore, CBR may find a longer but lightly loaded path better than the heavily loaded shortest path. Network traffic is thus distributed more evenly.

When MPLS and CBR are used together, they make each other more useful. MPLS's per-LSP statistics provide constraint based routing with precise information about the amount of traffic between every ingress-egress pair. CBR can be one of two types: online or offline. The online CBR allows the routers to compute paths for LSPs at any time. In the offline CBR, an offline server computes paths for LSPs periodically. LSPs are then configured to take the computed paths.

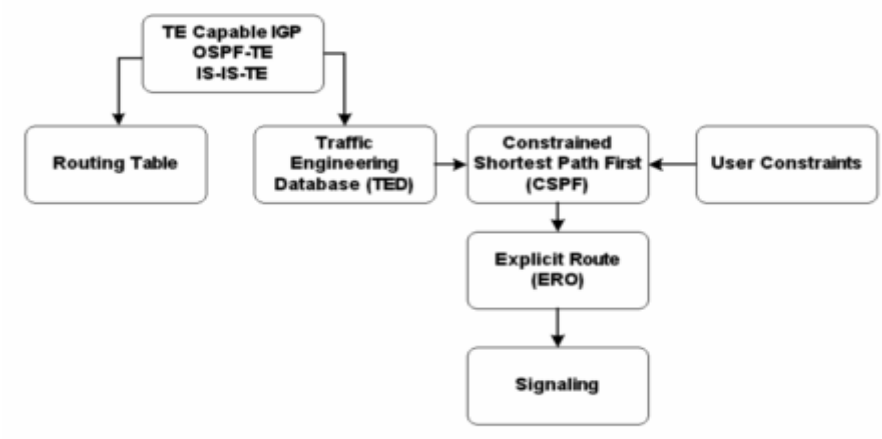


Fig.1.10 Procedure of Constraint-based routing in MPLS network

In TE, CBR is used to calculate traffic trunks based on the optimization objectives. To perform such optimization, the TE system often needs network-wide information on topology and traffic demands. Thus TE is typically confined to a single administrative domain.

MPLS and TE address the issues of bandwidth provisioning and performance optimization in Internet backbones. The explicit route mechanism in MPLS adds an important capability to the IP-based network. Combined with CBR, MPLS and TE can help network providers which make the best use of available resources and reduce costs.

1.3.3. Routing metrics and path computation

Routing metrics expresses one or more of the characteristics of the route in terms that can be used to evaluate the suitability of this route, compared to another route with

different properties, for conveying a particular packet of class of packets. They have major implications not only on the range of QoS requirements that can be supported but also on the complexity of path computation. In [Lepage, 2006], an architecture adapting the application to the actual QoS is designed.

1.3.3.1. Metrics

Routing algorithms determine that some routes are preferable to others by many different metrics. The metrics must reflect the basic network properties of interest. Sophisticated routing algorithms can base route selection on multiple metrics, combining them in a single (hybrid) metric. All the following metrics have been used:

- *Hop count* is the most common routing metric. A hop is a metric value used to measure distance based on the number of networks a datagram traverses. Each time a router forwards a datagram onto a segment this count as a single hop. Routing protocols such as RIP [RFC1058] or IGRP [Hedrick, 1991] that observe hops as their primary metric value consider the best or preferred path (when multiple paths exist) to a destination to be the one with the least number of network hops.

- *Bandwidth* refers to the available traffic capacity of a link. Bandwidth is measured in terms of bits per second (bps). These protocols determine and consider the bandwidth capacity of each link along the path end to end. The path with the overall higher bandwidth is chosen as the best route. Protocols such as OSPF [RFC2328] are based on bandwidth criteria.

- *Routing delay* refers to the amount of time required to send a packet from source to destination through network. Delay depends on many factors, including the bandwidth of intermediate network links, the port queues at each router along the way, network congestion on all intermediate network links, and the physical distance to be traveled. Protocols that use this metric must determine the delay values for all links along the path end to end, considering the path with the lowest (cumulative) delay to be a better route

- *Reliability*, in the context of routing algorithms, refers to the dependability of each network link. Some network links might go down more often than others. After a network fails, certain network links might be repaired more easily or more quickly than other links. Any reliability factors can be taken into account in the assignment of the reliability ratings, which are arbitrary numeric values usually assigned to network links by network administrators. Routers observe attached links, reporting problems, such as link failures, interface errors, lost packets and so on. Links experiencing more problems would be considered less reliable than others making them less desirable paths; the higher the reliability the better the path. Because network conditions are constantly changing, link reliability will change. This value is generally measured as a percentage of 255, with 255 being the most reliable and 1 being least reliable.

- *Load* refers to the degree to which a network resource, such as a router, is busy. Load is a variable value, generally measured over a five-second window indicating the traffic load over a specific link. Load measures the amount of traffic occupying the link over this time frame as a percentage of the link's total capacity. The value 255 is equivalent to 100% utilization or load; the higher this value is, the higher the traffic load (bandwidth utilization) across this link is. As traffic increases, this value augments. Values approaching 255 indicate congestion, while lower values indicate moderate traffic loads—the lower the value, the less congested the path, the more preferred. This value may be manually configured as a static value by an administrator or it may be dynamically tracked allowing it to adjust as traffic patterns within the network change.

- *Communication cost* is another important metric, especially because some companies may not care about performance as much as they care about operating costs. Although line delay may be longer, they will send packets over their own lines rather than through the public lines that cost money for usage time. Network administrators can affect the way routers make path decisions by setting arbitrary

metric values on links along the path end to end. These arbitrary values are typically single integers with lower values indicating better paths.

1.3.3.2. Routing decision with several metrics

Metric is a characterization parameter for a path or link in a network. A QoS metric is a metric that characterizes the quality of the path.

In order to find feasible paths that satisfy the QoS requirements of a flow, suitable metrics based on QoS parameters must be defined. The metrics have to be selected so that the requirement can be presented by one metric or reasonable combination of them. The path computation based on a metric or a combination of metrics must not be too complex as to make it impractical. As the metrics define the types of QoS guarantees the network is able to support, no requirement can be supported if it cannot be mapped onto a combination of the selected metrics. The metrics commonly used on QoS routing and CBR are divided into three categories, also called the composition rules of the metrics.

Let $m(i_1, i_2)$ be a QoS metric for link (i_1, i_2). For any path $P = (i_1, i_2, i_3 \cdots i_{n-1}, i_n)$, the metric is [Shigang, 1999]:

- *Additive* if $m(P) = m(i_1, i_2) + m(i_2, i_3) + \cdots + m(i_{n-1}, i_n)$

  For example, the delay and the hop-count are additive metrics.

- *Multiplicative* if $m(P) = m(i_1, i_2) \cdot m(i_2, i_3) \cdot \cdots \cdot m(i_{n-1}, i_n)$

  For example, reliability is a multiplicative metric.

- *Concave* if $m(P) = \min\{m(i_1, i_2), m(i_2, i_3), \cdots, m(i_{n-1}, i_n)\}$

  For example, bandwidth is a concave metric.

According to the type of the metric, different algorithms can be used to compute the candidate paths. For example, considering an additive metric such as hop-count, one generally uses the shortest path algorithm to retain some paths that are shorter than a given length. Bandwidth is an example of a concave metric, because on a path, the link with the smallest bandwidth determines the total available bandwidth on the path.

Propagation delay is additive because it is the result of the sum of all the propagation delays on each link.

No matter what path constraints are used, routers with constraint-based routing will have to compute its routing table more frequently. This is because, even without topology changes, routing table computation can still be triggered by significant resource availability changes or changes in link affinity. Therefore, even if the complexity of a CBR algorithm is comparable with the complexity of normal routing algorithm, the computation load on routers with CBR can be significantly higher. It can cause stability problem of the networks.

1.3.3.3. Need of heuristic approaches

Multiple metrics can provide more accurate information for routing decisions. But the problem is that finding a path subject to multiple constraints is not easy, and often impossible. It has been proved that finding an optimal path subject to constraints on two or more additive and/or multiplicative constraints in any possible combinations is NP-complete [Shigang, 1999]. As a result, any algorithms that select any two or more of delay, jitter, hop counts, error rate as criteria and tries to optimize them simultaneously is NP-complete. The computationally feasible combinations of metrics are bandwidth and one of the rests (delay, delay jitter, hop-count, cost, reliability).

A lot of heuristic algorithms are proposed to solve this problem. A common method is called “sequential filtering”, under which a combination of metrics is ordered in some fashion, reflecting the importance of different metrics. Paths based on the primary metric are computed first and a subset of them are eliminated based on the secondary metric and so on, until a single path is found. This is tradeoff between performance optimization and computation simplicity.

As examples, SWP (Shortest Widest Path) [Wang, 1996] and WSP (Widest Shortest Path) [Guerin, 1997] are sequential filtering algorithms. SWP chooses feasible paths with maximum bandwidth and, if there are multiple such paths, the one with the shortest hop count. SWP can use much more resources since it select the path with maximum bandwidth instead of selecting shorter path with sufficient bandwidth to treat traffic.

The WSP algorithm is an improvement of the Min-Hop algorithm, as it attempts to load-balance the network traffic. In fact, WSP chooses a feasible path with minimum hop count and, if there are multiple such paths, the one with the largest residual bandwidth, thus discouraging the use of already heavily loaded links. However, WSP has the same drawbacks as Min-Hop algorithm since the path selection is performed among the shortest feasible paths that are used until saturation before switching to other feasible paths.

SWP and WSP are for calculating the best one path. Other multipath routing algorithms can use more than one criterion simultaneously for selecting candidate paths in a polynomial time or higher complexity. And multipath routing algorithms decide the cost function by link or by path, with one or more criteria. In next chapter, we will introduce multipath routing scheme and the cost function.

1.4. Summary of Contribution

IETF proposed some QoS architecture. The major schemes are IntServ and DiffServ that provide new architecture for resource allocation in the Internet.

IntServ uses RSVP signaling protocol which is used for resource allocation and admission control for traffic. But it needs so much state information for individual flow that it is not scalable and not suitable for large scaled network.

The DiffServ architecture is composed of a number of functional elements implemented in network nodes to provide per-hop forwarding behaviors, packet classification function, and traffic conditioning functions such as metering, marking, shaping and policing. This architecture achieves scalability by implementing complex classification and conditioning functions only at network boundary nodes, and by applying PHB to aggregates of traffic which permits a reasonably granular means of allocating buffer and bandwidth resources at each node among competing traffic streams. The intermediate routers do forwarding path decisions separately and more simply than the setting up of the service agreements and traffic profile.

TE is concerned with the performance optimization of operational networks. Its main objective is to reduce congestion hot spots and improve resource utilization across the network through carefully managing the traffic distribution inside a network. For the sake of more efficient TE in IP networks, network administrators must be able to control the paths of packets. It needs some kinds of connections in the connectionless IP networks. MPLS can support multiple protocols and do fast transmission and can provide the connections with LSPs by explicit route. This explicit route specially is a basic mechanism for facilitating TE and one of the reasons why MPLS is useful for IP networks. MPLS and TE address the issues of bandwidth provisioning and performance optimization in Internet backbones. Combined with constraint-based routing in TE, MPLS and TE can help network providers make the best use of available resources and reduce costs.

The purpose of TE is to increase service availability and to reduce the load of each router. Multipath routing is the one of the mechanisms for load balancing in which the total load from a source to a destination is spatially distributed over several paths.

In the next chapter, the general multipath routing scheme and some algorithms in the literature will be presented.

Chapter 2:

Multipath routing in MPLS network

2.1. Introduction

In overlay approach [RFC2917], service providers establish logical connections between the edge nodes of a backbone, and then overlay these logical connections onto the physical topology. The overlay approach therefore replaces the hop-by-hop routing paradigm using shortest paths in traditional IP networks with explicit routing via which logical connections use any one of the feasible paths through the network. MPLS technology provides mechanisms in IP backbones for explicit routing to facilitate traffic engineering. In MPLS network, a constraint-based routing scheme can also be used so that traffic may be controlled to flow optimally through certain routes.

Multipath routing in MPLS network routes demands on multiple paths simultaneously for balancing the load in the network using explicit routing and constraint based routing instead of routing all the traffic on the only one shortest path.

There are 4 types according to the combination of the number of parameters to estimate the paths' quality and the number of feasible paths to assign current demand; single path with single parameter, single path with multiple parameters, multiple paths with single parameter and multiple paths with multiple parameters. Their representative algorithms in the literature are as follows respectively.

- Single path with single parameter: Dijkstra's shortest path algorithm [Dijkstra, 1959].
- Single path with multiple parameters: Shortest Widest Path [Wang, 1996], Widest Shortest Path [Guerin, 1997].

- Multiple paths with single parameter: Equal Cost Multi Paths [RFC2328] [RFC2992].
- Multiple paths with multiple parameters: Load Distribution in MPLS [Song, 2003], Widest Disjoint Paths [Srihari, 2001], MATE [Elwalid, 2001], AIMD [Wang, 2002].

Today's Internet provides only a single path between any pair of hosts that fundamentally limits the throughput achievable between them. For example, very large scaled and higher rate needed traffic in the multimedia application like video conferencing or IP telephony can experience congestion if they are sent through only a single path. For these multimedia applications, if multiple transport level flows would be mapped to different paths in the network, and they have control over splitting traffic between these flows, these applications could be sent more efficiently having less congestion probability.

Multipath routing algorithm selects feasible paths using one or more criteria defining a metric and then distributes the traffic demand among selected paths. Multipath routing can be effectively used for maximum utilization of network resources and it is capable of aggregating the resources of multiple paths and reducing the blocking capabilities in QoS oriented networks, allowing data transfer at higher rate when compared to single path.

In this paper, Multipath routing with single or multiple parameters in IP-based MPLS network is focused.

The various algorithms in the literature give solutions for effectively computing the multiple paths and ways to minimize delay and increase throughput. In this chapter, first of all, two path characteristics that allow applications to increase their performance, path quantity and path independence, are presented. Then the general schematic approach of multipath routing algorithm is proposed and the problem formulation of traffic bifurcation is identified. Some existing multipath algorithms are then compared in order to verify if they are scalable and efficient for using in the big network.

2.2. Path Characteristics

For multipath routing, multiple paths set to assign the current demand must be calculated first. The quality of a path set depends on the intended use of the paths in the set. Although path set quality depends on its intended use, normally there are some path characteristics that allow applications to increase their performances.

Two such path characteristics are described, path quantity and path independence. Path quantity is the number of paths calculated between nodes. Path independence refers to the disjointedness of the calculated paths. These two characteristics are based on the increase of end-to-end performance by providing multiple paths between nodes.

2.2.1. Path Quantity [Chen, 1999]

There are two ways to describe path quantity. One method is to consider the total number of paths calculated in a path set. The multipath network means the one which allows applications to use the multiple paths for sending their traffic. The fundamental difference between a multipath network and a single path network is the number of paths provided between nodes. With more paths, nodes have more options to interact with other nodes, potentially increasing their performance. Thus, when the number of paths is higher, the chances for load distribution are better. But, it is desirable to use as few paths as possible while at the same time minimize the congestion in the network. Because there is a significant overhead associated with establishing, maintaining and tearing down of paths and the complexity of the scheme that distributes traffic among multiple paths increases considerably as the number of paths increases [Srihari, 2001].

Notably [Leung, 2000] proved that the number of paths used in multipath routing should be small, say up to three.

2.2.2. Path Independence [Chen, 1999]

The second path characteristic, path independence, describes the disjointedness of the feasible paths provided between nodes. This property is important because, the more independent a path set is, the more aggregate physical resources the set offers between a

node pair as those resources are not shared, and the less likely the performance of one path affects the performances of other paths.

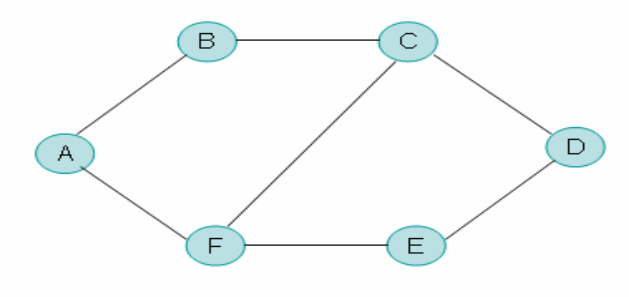


Fig. 2.1 Path Independence

Fig.2.1 is the network to illustrate the importance of path independence. Consider a path set with 2 paths (A, B, C, D) and (A, F, C, D) and other path set with 2 paths (A, B, C, D) and (A, F, E, D). The second set is independent when compared to the first set. So the second set would lead to better usage of resources and is less likely to be congested because at least one link in each path should be congested, whereas in the first set congestion at link (C, D) is reflected in both the two paths. Multipath sets with this attribute facilitate higher performance.

In addition to increasing performance, path independence also allows end-hosts to achieve higher network availability. In the case of link failure, an end-host has a higher probability of reestablishing connectivity on a different path if the paths provided are independent. In contrast, in a single path environment, where there is no path independence, an end-host must wait for the routing algorithm to detect the failure and to re-compute the appropriate paths in order for the end-host to reestablish connectivity.

In summary, multipath sets with these characteristics, path quantity and path independence, better allow end-hosts to increase their performance, compared with path sets that do not consider these features.

2.3. Multipath routing for Load Balancing

Multipath routing is a load balancing technique in which the total load from a source to a destination is spatially distributed over several paths. The multipath routing model

offers applications the ability to increase their network performance and to minimize network delays by balancing the network load on a set of network paths. With more stable delays and balanced utilizations experienced over network links, these multipath routing protocols are relatively more stable to both changing network conditions and traffic burstiness.

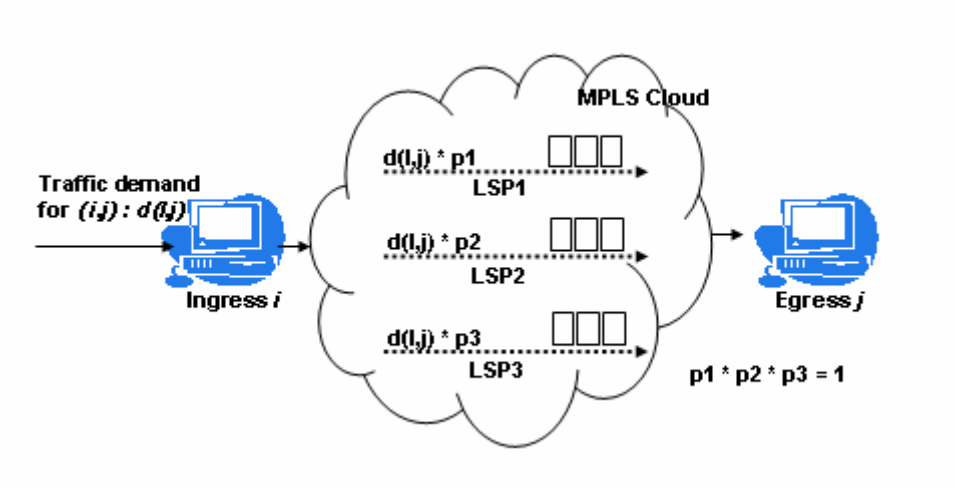


Fig.2.2 Multipath routing

Traffic engineering in IP networks uses the traffic trunk concept [Daniel, 1999]]. A traffic trunk is an aggregation of traffic belonging to the same class. It is essentially an abstract description of traffic that allows certain attributes of the traffic to be parameterized. It is independent of the underlying technologies. The problem of mapping traffic trunks onto a given network topology is a certain issue of traffic engineering. In MPLS networks, traffic trunks are mapped onto the network topology through the selection of routes for explicit LSPs. The terms LSP tunnel and traffic engineering tunnel are commonly used to refer to the combination of traffic trunk and explicit LSPs in MPLS.

LSP tunnels allow the performance of the operational network to be optimized in various ways. For example, if congestion problem caused by routing on the inefficient route or by traffic concentration on a route are detected, LSP tunnels can be rerouted to solve the problem. LSP tunnels can be parameterized, and network resources can be allocated to them based on those parameters. Multiple LSP tunnels can be created between two nodes, and the traffic between nodes divided among the tunnels according

to some administrative policy. LSP tunnels permit the introduction of flexible and cost-effective competitive options. Statistics derived from LSP tunnels can be used to construct an elementary traffic matrix [Daniel, 1999].

2.3.1. LSP calculation mode

MPLS can help introduce the required TE (Traffic Engineering) functionalities, while avoiding the problem of route oscillation and route cycles. The fact that all packets labeled with the same FEC (Forwarding Equivalence Class) are guaranteed to be delivered over the same route which allows us to define a virtual topology constituted by the LSPs mapped onto the physical topology. There are two ways in which the optimal virtual layout can be calculated, online and offline.

2.3.1.1. Offline mode

In the offline approach, the source maintains a number of multiple pre-established paths to the destination and routes the traffic demand along those paths. Route computation is performed for all routes periodically with current information, and the network cuts over to the new routes during maintenance periods. This approach tends to yield optimal results since all routes are systematically re-optimized after changes. However, frequent and large scale rerouting of existence of connections will cause disruption to traffic flows; thus it may be undesirable from the operational point of view. In addition, extra delays result from adding new traffic demands to the network as route computation is performed periodically.

2.3.1.2. Online mode

In the online approach, the source computes dynamically feasible paths for each individual arrival request, which is known as constraint-based-routing. When a new demand arrives, the route computation module calculates an optimal route for the new demand only, without changing the routes for exist demands and routing algorithms monitor link costs and re-compute paths upon detection of link cost changes. Therefore rerouting to exist traffic flows is minimized, although the resource utilization may not

be as efficient as that in the offline approach. But the benefits of on-demand route computation and minimum impacts on existing traffic often outweigh the loss in efficiency.

The two modes may be combined at different time scale. For example, the routes for new demands can be placed incrementally, and after some time interval complete re-optimization is performed for all demands during less busy periods.

2.3.2. Schematic approach of Multipath routing [Lee, 2005]

Routing control in traffic engineering essentially balances traffic load over multiple paths in the network to minimize congestion. It concerns how to select paths and to distribute traffic among those paths such that given QoS constraints are met or close to the target at a large possible extent while at the same time optimizing some global network-wide objectives such as utilization.

In cases where traffic demands have other QoS or administrative constraints, the global objective becomes minimizing the rejection rate for demand or blocking rate. But at the moment, most existing works in the literature only consider bandwidth or hop-count as the QoS constraints.

To allow comparing the different approaches, we propose first a generic framework of multipath routing.

There are basically two main steps in a multipath routing algorithm: *computation of multiple paths and traffic splitting among these multiple paths.* In a multipath network, the time and calculation complexities required to support these two steps are notably higher than those in single path routing [Chen, 1999].

2.3.2.1. Step1 : Candidate Paths Selection

As depicted in Fig.2.3, a multipath routing algorithm first needs to calculate the paths it wishes to provide. The first step computes the set of candidate paths. This set is a subset of all the paths between the pair of considered routers. These candidate paths are defined with respect to a cost function. According to the nature of the cost function, different algorithms can be applied to determine these candidate paths. To specify the

cost function, the authors consider various criteria such as the bandwidth, hop count, delay, error rate, and so on. Generally this step uses static criteria. So, the first problem in this step can be summarized by finding the appropriate cost function or metric that can take some of the previous criteria into account. We generalize this problem by the following equation:

$$\textit{Candidate path set} = f(\textit{bandwidth, hop-count, delay, error rate}) \qquad (2.1)$$

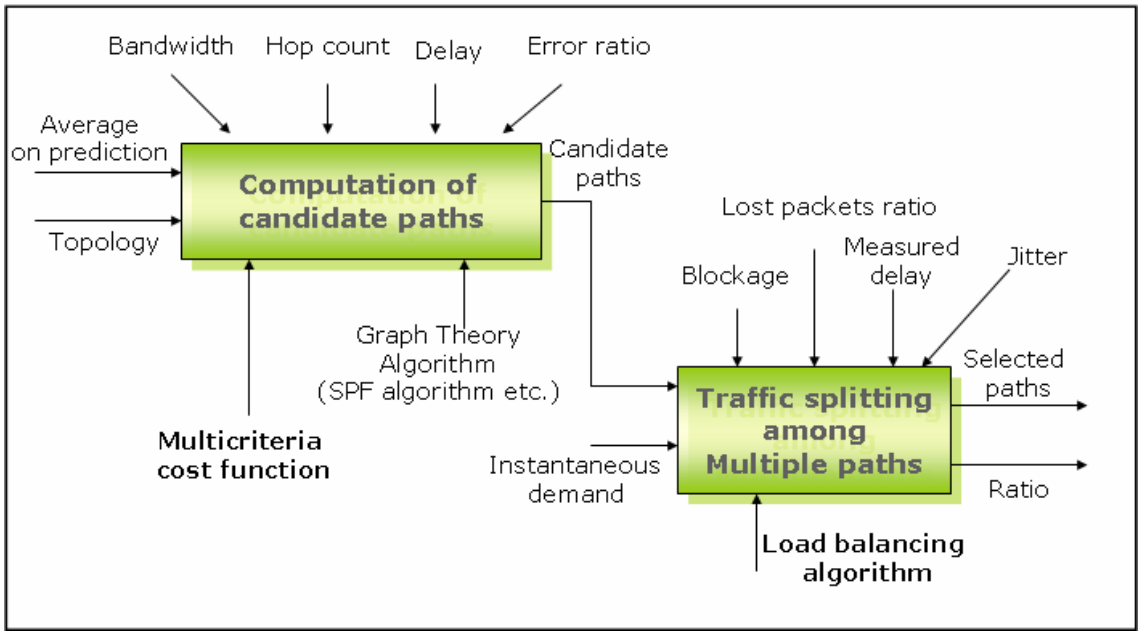


Fig.2.3 Functional decomposition of multipath algorithms

One of the ways end-hosts can benefit from a multipath network is to reduce their communication delays to other nodes. For example, because VoIP traffic is sensitive to delay, it calculates not high throughput paths but low-delayed paths. In order to calculate paths that minimize delay, a path calculation algorithm needs to measure the expected delay of a link. In naïve approaches, a path's delay is generally determined by the number of hops. For the low delay path service, the objective is to calculate paths so that the paths' delay between a node pair is minimized. As a path's actual delay is traffic dependent, a multipath calculation algorithm needs to calculate paths so that low delay can be obtained under different traffic patterns. Algorithms for obtaining the paths

with minimum delay are Shortest K Paths [Eppstein, 1998], Link Disjoint paths [Oki, 1995], Discount shortest paths [Palakurthi, 2001] and so on.

Throughput oriented applications such as FTP calculate multiple paths which can increase their effective throughput. Again, the amount of throughput an application obtains depends not only on the paths calculated, but also on network traffic patterns. Thus, the paths calculated should be robust in the sense that they provide high throughput under a variety of traffic conditions. Normally, an expected link bandwidth can be characterized by a link capacity metric; when the link capacity is higher, the expected available bandwidth of the link becomes higher. Algorithms for obtaining the paths with high throughput are Maximum flow path [Cormen, 1990], Capacity removal Algorithm [Palakurthi, 2001] and so on.

In this candidate path selection step, different cost function can be used according to TE objectives of various applications and administrative policies.

Most of the time, the final objective of authors is to select some candidate paths to optimize some criteria depending on the QoS requirements of a user application. In this study, our aim is not to optimize individually different QoS requirements. We want to perform a global optimization of network QoS in order to give a QoS that is very close of the optimum that can be obtained by an individual optimization, to each user application. In this context, the problem of a cost function definition is typically a multi-criteria problem [T'kindt, 2002]. This multi-criteria problem is an NP-complete problem and consequently is difficult to solve (see section 1.3.3.2). In chapter 1, we have explained why the computation of several criteria is generally difficult and that we need heuristic to define cost functions that integrate several QoS criteria. It is why most of the approaches that are proposed today are mono criterion (case of individual QoS requirement optimization) or hierarchical criteria approaches.

SWP (Shortest Widest Path) [Wang, 1996] and WSP (Widest Shortest Path) [Guerin, 1997] are examples of algorithms based on hierarchical criteria. In the section 1.3.3.3 of chapter 1, we have shown that although they both use the same criteria (bandwidth and hop-count), they do not get necessarily the same candidate paths for a given network. This result proves clearly a hierarchical criteria system can perform

individual optimization of flows' requirements but not a global optimization. So a main goal of this study is to propose an approach to obtain a global optimization.

2.3.2.2. Step2 : Distributing traffic over multiple paths

The second step of multipath routing algorithms consists in splitting traffic among multiple candidate paths. The paths for this step are qualified of candidate paths because they are not necessary all used at each time. Traffic splitting ratio are decided depending on the evaluation of dynamic criteria such as blockages, packet loss rate, measured delay, jitter and so on.

In this case, the problem remains the same as the first step: how to define a cost function or metric that combines different previous criteria.

Traffic Splitting ratio = g (blockages, packets lost rate, measured delay, jitter) (2.2)

The traffic splitter takes incoming packets and distributes them to multiple outgoing links. The traffic may be split equally or in some specified proportion among all outgoing links. Routers can forward packets either in packet level or flow level mode.

- **Packet-level Distribution**

A router employing multipath routing protocol distributes incoming packets for the same flow to one of multiple next-hops in round-robin or random manner. Although this mechanism can be easily implemented, it will cause many out-of-order packet arrivals at the destination because multiple paths may have different delays.

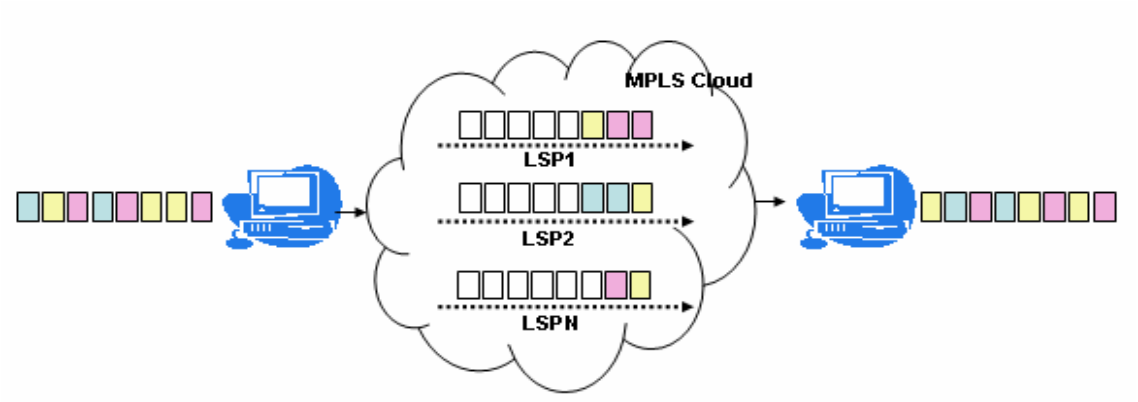


Fig.2.4 Packet level multipath forwarding

Although multipath routing has lots of benefits compared to single path routing, it is known that one of the biggest barriers to deploy multipath routing schemes is TCP's low performance in the presence of persistent packet reordering [Bohacek, 2003].

TCP performance over the multiple paths is supposed to be enhanced, because the bottleneck bandwidth increases. However, TCP senders will regard triple duplicate acknowledgments (ACK), resulting from out-of-order packet arrivals at the receiver due to the different delays of the multiple paths, as an indicator of a packet loss, and will retransmit the lost packet (Fast Retransmission). Not only this wastes extra bandwidth, but also it reduces the congestion window (*cwnd*) of the TCP sender (Fast Recovery). Moreover, if ACKs arrive at the TCP sender out-of-order, the late segment with the lower sequence number will be ignored and cannot serve for increasing the *cwnd*, since TCP senders accept only the in-order ACKs. Besides the variable path Maximum Transfer Unit (MTU) size for each packet will make path MTU discovery useless. The response time of very large flows depends more on TCP congestion avoidance than on the initial slow start phase.

As long as the packet loss rate p is not too high and the receive window is not limiting, the throughput B achieved by a permanent flow is given by the approximate relation:

$$B(p) \approx \frac{K}{RTT\sqrt{p}} \tag{2.3}$$

where K is a constant that depends on second-order statistics of the loss process (K = $\sqrt{2/3}$ for periodic losses) [Fredj, 2001b].

This will degrade end-to-end performance while the buffer requirement is increased [Padhye, 1998].

- **Flow-level Distribution**

It is well known that Internet traffic at packet level is extremely variable over a wide range of time scales. This variability is manifested by asymptotic self-similarity and multi-fractal behavior at time scales smaller than that of the round trip time. A plausible explanation for self-similarity is the heavy-tailed nature of the size distribution of

transferred documents while multi-fractal behavior appears to be a manifestation of the burstiness induced by TCP congestion control [Fredj, 2001a].

The complexity of the packet arrival process is such that it proves very difficult to derive a packet level traffic characterization which is useful for performance modeling [Fredj, 2001a]. Therefore, for traffic control purpose, it is natural to characterize traffic at the flow level instead. A flow in the Internet is defined as a succession of packets near in time to one another, generated by the same application instance between a particular source and destination pair. Flows may be broadly categorized as streaming or elastic according to their nature. Stream flows are representative of real time applications, audio or video. They have intrinsic temporal properties that the network must preserve. The elastic flows are representative of file transfer-like applications. For elastic flows the transfer throughput depends on the available bandwidth, the transfer duration is a function of the file size and traffic characteristics of the transport network during the transfer. Elastic traffic is the most important of the Internet in terms of its proportion to stream traffic. To describe the flow arrival process, the usual practice is to characterize it as a Poisson process.

Flow-level forwarding routers deliver packets which belong to a flow to the same next-hop. Routers classify packets streams into flows using flow detectors. However these schemes require per-flow states in routers, what is not always scalable.

The direct advantage of per-flow traffic distribution is that it can avoid the disorder problem that the per-packet mechanism has. This is due to the fact that, in the per-flow traffic distribution, all packets in a flow will go along the same path as long as the path is determined.

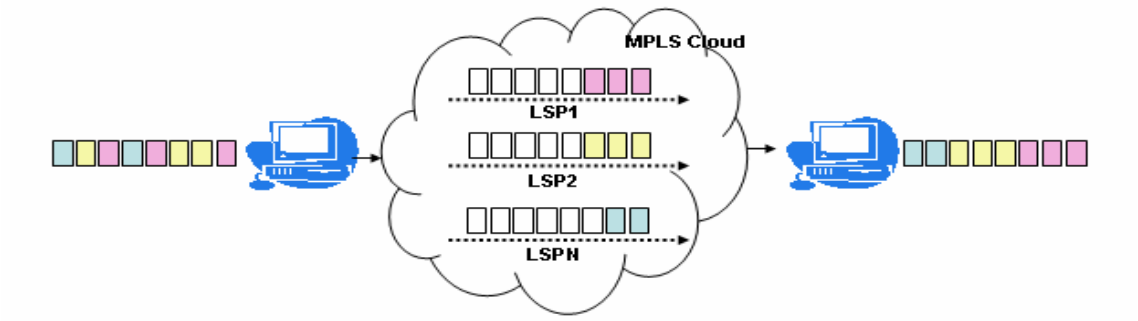


Fig.2.5 Flow level multipath forwarding

In the rest of this work, my proposition will be based on flow distribution in order to preserve actual TCP model.

2.4.Traffic Bifurcation

Heuristic approaches such as filtering approach (see section 1.3.3.3) cannot ensure that all QoS requirements can be guaranteed to all types of traffic. To improve IP routing, it is better to mix both load balancing and DiffServ prioritizing approach. Integration of load balancing and DiffServ will be treated in the chapter 4.

Traditional optimal routing [Bertsekas, 1992] permits traffic bifurcation to multiple paths. Traffic engineering is performed on a set of newly arrived LSP connection requests for the existing network, or periodically on all the established LSPs for the known demand. The results of the computations are paths to set up as LSPs and load-splitting ratios among the paths. The traffic bifurcation function for the given load-splitting ratio is handled by multipath forwarding MPLS routers, and multiple LSPs between an ingress router and an egress router are explicitly established by a signaling protocol.

When one studies some multipath routing algorithms, the optimal value by Linear Programming Formulation is needed as a basis to determine whether that algorithm is efficient for load balancing in a large network. Mathematical formulation of route optimization problem and basic solution to the problem will be described in this part.

2.4.1. Performance Criterion : Maximum link utilization

Although the problem of inadequate network resources must be solved with new capacity or by reducing and controlling the demands, unbalanced traffic distribution can be solved through appropriate traffic engineering policy. With carefully arranged traffic trunks, service providers can spread traffic across the network to avoid hot spots in parts of the network. A traffic engineering problem in the Internet regarding performance optimization has a focus on setting up paths between the edges routers in the network to meet traffic demands while achieving low congestion and optimizing the utilization of

network resources. In practice, the typical key objective of traffic engineering is to minimize the utilization of the most heavily used link in the network. Link utilization (U_l) is calculated as follows.

$$U(l) = \frac{c_l - r_l}{c_l} \quad \text{with} \quad 0 \le r_l \le c_l \tag{2.4}$$

- c_l : capacity of link l
- r_l : residual bandwidth of link l

When we translate this into a mathematical formulation, the objective is in essence to minimize the maximum link utilization in a network. Intuitively the hot spots are the points with the highest link utilization. Reducing the link utilization at these points balances the traffic distribution across the network.

This optimization objective of minimizing the maximum link utilization has a number of desirable features.

First, when the maximum link utilization minimizes, congestion rate is reduced and at the same time the total delay experienced by the packets reduced. Similarly total losses are minimized. Packet loss rate is useful as another performance metric.

Second, this optimization objective moves traffic away from congested hot spots to less utilized parts of the network, so that the distribution of traffic tends to be balanced and the network can achieve the load balancing effect.

Finally, it also leaves more space for future traffic growth. When the maximum of link utilization is minimized, the percentage of the residual bandwidth on links is also maximized. The growth in traffic therefore is more likely to be accommodated and can be accepted without requiring the rearrangement of connections. When the maximum link utilization minimizes, naturally the rejection rate for demand can be reduced. If we assume that traffic grows in proportion to the current traffic pattern, this objective will ensure that the extra traffic causes minimum congestion [Zhang, 2001].

2.4.2. Problem formulation of Traffic Bifurcation [Lee, 2002]

The traffic bifurcation problem (TB hereafter) consists of finding multiple paths carrying each part of or all the traffic between ingress and egress node which minimize

the maximum link utilization α. When splitting a traffic demand to multiple paths, the granularity of load splitting, g ($0<g\leq 1$) is defined to represent how roughly a traffic demand can be divided.

The network is modelled as a directed graph, $G = (V, E)$, where V is the set of nodes and E represents the set of links. The capacity of a directional link $(i,j) \in E$ is c_{ij}. K is a set of all LSP connection requests. Each traffic demand $k \in K$ is given for a node pair between an ingress router s_k and an egress router t_k. d_k is the amount of a traffic demand for the LSP connection request k and d_k is a scaling factor to normalize total traffic demand from the source to become 1. The variable X^k_{ij} represents the fraction of the traffic demand assigned to link (i,j) for the LSP connection request, k. The integer variable M^k_{ij} represents how many units of basic discrete split demands for a traffic demand k are assigned to link (i,j).

A mixed-integer programming can be used to formulate the discrete optimization problem. Mixed-integer programming is the formulation that some of the decision variables are constrained to have only integer values at the optimal solution and it is the minimization or maximization of a linear function subject to constraint. Mixed integer programming problem for the traffic bifurcation case is given by the equations (2.5) to (2.9). It consists in finding the optimal α and in identifying the bottleneck link.

Minimize α

Subject to

$$\sum_{j:(i,j)\in E} X^k_{ij} - \sum_{j:(i,j)\in E} X^k_{ji} = 0, \quad i \neq s_k, t_k \tag{2.5}$$

$$\sum_{j:(i,j)\in E} X^k_{ij} - \sum_{j:(i,j)\in E} X^k_{ji} = 1, \quad i = s_k \tag{2.6}$$

$$\sum_{j:(i,j)\in E} X^k_{ij} - \sum_{j:(i,j)\in E} X^k_{ji} = -1, \quad i = t_k \tag{2.7}$$

$$\sum_{k\in K} d_k \cdot X^k_{ij} \leq c_{ij} \cdot \alpha, \quad (i,j) \in E \tag{2.8}$$

$$X^k_{ij} = M^k_{ij} \cdot g \tag{2.9}$$

$$0 \leq X^k_{ij} \leq 1, \quad 0 \leq g \leq 1, \quad 0 \leq \alpha, \quad M^k_{ij} \in Z, \quad 0 \leq M^k_{ij} \leq \lfloor 1/g \rfloor$$

The objective is to minimize α. Equation (2.5), (2.6) and (2.7) represent the flow constraints for intermediate, source, and sink nodes, respectively. Equation (2.8) is the link capacity constraint. Equation (2.9) states that the fraction of the demand *k* assigned to a link is proportional to the number of units of the demand that have been assigned to the link (*i,j*) and that assigned traffic demands are discrete.

The *TB*(*g*) mixed-integer programming problem can be solved by searching the *branch and bound* tree with an MIP solver such as MATLAB [MATLAB] or CPLEX [CPLEX], and the solution gives the optimal flow values, X^k_{ij}. Especially, when *g* approaches to zero, the *TB*(*g*) program is reduced to an LP problem which can be solved with the classic Simplex Method. Moreover, if *g* is 1, the *TB*(*g*) problem is the integer programming problem for the non-bifurcation case.

In a realistic environment, these equations are generally not satisfied. So, heuristics are needed to obtain a reasonably good solution close to the optimal one instead of solving the above optimization models directly. From the operational perspective, it is certainly more desirable to have each demand routed over a single path, even with less optimal performance. The mathematical formulation for the case where each demand must be routed over a single path is quite straightforward. We can use the same formulation as in linear programming formulation but with the additional restriction that the X^k_{ij} variables must be either 0 or 1, so that no fraction of a demand can be put on another route. This optimization problem then becomes an integer-programming one. It can be shown that the problem is NP-hard, so no efficient algorithms exist that provide exact solutions. Nevertheless, heuristic solutions exist that provide sufficiently good results to the problem [Bertsekas, 1992]. In the rest of this chapter, some multipath routing algorithms proposed in literature are described and estimated whether they are efficient and stable or not.

2.5. Complexity study of various MPLS-based Multipath routing algorithms [Lee, 2005a] [Lee, 2006b]

Multipath routing has been extensively researched for over two decades in circuit-switched networks. In literature, there are a lot of algorithms proposed for multipath routing. Each algorithm is declared very efficient by its authors but generally with respect to restricted conditions. Since all actual propositions depend on specific conditions, there can be the real problem that is not to define a unique routing model. Specifically this subchapter compares some recent algorithms such as MATE [Elwalid, 2001] or LDM [Song, 2003] with regard to the generic framework of multipath routing algorithm proposed in section 2.3. The comparison is also done with regard to the scalability and the stability of each solution in order to verify if they are scalable and efficient for a large network.

2.5.1. WDP (Widest Disjoint Path) [Srihari, 2001]

2.5.1.1. Step1 : Candidate Paths Selection

WDP (Widest Disjoint Paths) is not a full multipath-routing algorithm, but focuses on the selection of good disjoint paths. As discussed in 2.3, the path disjointedness characteristic allows end hosts to increase their performance. This approach is mainly based on two concepts: path width and path distance. Path width is the residual bandwidth on its bottleneck link. More specifically, if we consider a path composed by different links l and if c_l is its residual bandwidth and $\hat{c}_l$ is its bandwidth capacity, equation (2.10) defines the width W of a path:

$$W = \min_{(l)\in path}(c_l) \quad \text{with} \quad 0 \le c_l \le \hat{c}_l \tag{2.10}$$

Using this path width concept, bottlenecks in the network can be detected and avoided if possible.

$$d = \sum_{l \in path} \frac{1}{c_l} \tag{2.11}$$

Path distance is calculated as the sum of the reciprocals of the average residual bandwidth on each link defining the path (equation 2.11). The shorter the distance, the better is the path. Path distance is original because contrary to most approaches, it is not a hop-count measure but it is indirectly dependent on the utilization ratio of each link defining the path. The idea is that the residual bandwidth reflects the congestion of the pair of nodes of the link. Consequently, if the residual bandwidth is near 0 this means that the congestion probability is high and then the transmission delays increase. In this way, WDP is an improvement of SWP (Shortest Widest Path) algorithm [Wang, 1996].

This approach is very promising when considering a practical implementation with numerous ingress-egress router pairs but normally the link disjoint algorithm achieves high path independence at the price of path delay. Consequently it lowers the probability that end-hosts can obtain low delay paths to their destinations but at the same time it lowers the congestion probability owing to its disjointedness.

WDP algorithm performs candidate paths selection based on the computation of the width of the good disjoint paths with regard to bottleneck links. It selects a restricted number (η) of paths fixed by administrator (see Fig. 2.6). A path is added to the subset of good paths if its inclusion increases the width of this subset. At the opposite, a path is deleted if its deletion does not reduce the width of the subset of good paths. This is a heavy computation to perform on every path: computing a set of n paths takes $O(n^2)$ cycles because the selection procedure proposed in Fig.2.6 is clearly in $O(n^2)$ and allows selecting one path at each iteration considering all potential paths between the pair of ingress-egress routers. But, WDP algorithm makes the path set independent so it allows end-hosts to achieve higher network availability. If WDP is used jointly in offline and online approach together but appropriately, its real-time complexity can be reduced more and less. For example, there can be the algorithm that link disjoint paths are pre-calculated in the offline mode and then it calculates the widest disjoint paths among them with dynamic link utilization rate in online mode.

The narrowest path is the last path removed from the set R and referred to as NARROWEST(R). Fig. 2.6 depicts the procedure of candidate paths selection for ingress-egress pair σ.

In the algorithm given by Fig. 2.6, let us consider the following notations:

- R_σ : the candidate path set for a pair σ
- $\hat{R}_\sigma$: the set of feasible paths for routing flows between the pair σ
- R : the width for a set of paths
- v_r : the offered load on path r
- b_r : the blocking probability on path r
- c_l : the average residual bandwidth on link l
- η : the specified limit number of candidate paths

```
PROCEDURE SELECT(σ)
    For each path r in R_σ
        For each link l in r
            c_l = c_l - ( 1 - b_r ) v_r
    If |R_σ| < η
        W_r = WIDTH(R_σ ∪ r), ∀r ∈ R̂_σ \ R_σ
    Else
        W_r = WIDTH(R_σ ∪ r \ NARROWEST(R_σ ∪ r)), ∀r ∈ R̂_σ \ R_σ
    W+ = max_{r∈R̂σ \ Rσ} W_r
    If ( W+ > ( 1+ ψ)WIDTH(R_σ))
        R+ = { r : r ∈ R̂_σ \ R_σ, W_r = W+ }
        d+ = min_{r∈R+} d_r
        r+ = { r : r ∈ R+, d_r=d+ }
        r- = NARROWEST (R_σ ∪ r)
        If (|R_σ| = η or WIDTH ( R_σ ∪ r+ \ r-) = W+)
            R_σ = R_σ ∪ r+ \ r-
        Else
            R_σ= R_σ ∪ r+
        Else
        r- = NARROWEST(R_σ)
        If WIDTH (R_σ\ r-)=WIDTH(R_σ)
            R_σ = R_σ \ r-
END PROCEDURE
```

Fig.2.6 The candidate path set selection procedure for pair σ

We have modified some parts of this algorithm in Annex A.

2.5.1.2. Step2 : Distributing traffic over multiple paths

For the traffic splitting stage, the authors of study propose *ebp* (equalizing blocking probability) or *ebr* (equalizing blocking rate) that are localized approaches.

Let $\{r_1, r_2, ..., r_k\}$ be the set of k candidate paths between a source destination pair of nodes. The objective of the *ebp* strategy is to find a set of proportions $\{\alpha_{r1}, \alpha_{r2, ...} \alpha_{rk}\}$ such that flow blocking probabilities on all the paths are equalized, i.e., $b_{r1} = b_{r2} = \cdots = b_{rk}$, where b_{ri} is the flow blocking probability on path r_i. On the other hand, the objective of the *ebr* strategy is to equalize the flow blocking rates, i.e., $\alpha_{r1}b_{r1} = \alpha_{r2}b_{r2} = \cdots = \alpha_{rk}b_{rk}$. By employing these strategies a source node can adaptively route flows among multiple paths to a destination, in proportions that are commensurate with the perceived qualities of these paths. The perceived quality of a path between a source and a destination is inferred based on locally collected flow statistics: the offered load on the path and the resulting blocking probability of the flows routed along the path.

In this work, the authors use a simpler approximation of *ebp* that computes new proportions as follows. First, the current average blocking probability $\bar{b} = \sum_{i=1}^{k} \alpha_{ri} b_{ri}$ is computed. α_{ri} is the proportion of path *ri*. Then, the proportion of load onto a path r_i is decreased if its current blocking probability b_{ri} is higher than the average $\bar{b}$ and increased if b_{ri} is lower than $\bar{b}$. The magnitude of change is determined based on the relative distance of b_{ri} from $\bar{b}$ and some configurable parameters to ensure that the change is gradual.

The complexity of this algorithm is $O(n)$ since it consists in updating the blocking probability b_{ri} of each path r_i depending on the relative distance of b_{ri} from the current average blocking probability $\bar{b}$.

In the next chapter, only WDP candidate path selection part of this paper is used for the hybrid schemes.

2.5.2. MATE (MPLS Adaptive Traffic Engineering) [Elwalid, 2001]

2.5.2.1. Step1 : Candidate Paths Selection

MATE paper describes a multipath adaptive traffic engineering scheme targeted for MPLS networks. Contrary to other algorithms compared here, there is no procedure to

select candidate paths in MATE. So all available paths are considered and it does not impose any particular scheduling, buffer management, or a priori traffic characterization on the nodes. Anyway, another step is required before proceeding to the load distribution: the incoming packets are regrouped into a fixed number of bins.

2.5.2.2. Step2 : Distributing traffic over multiple paths

The number of bins determines the minimum amount of data that can be shifted. There are several methods to distribute the traffic between the bins. They differ in their complexity and the way they handle packet sequencing:

- *Per-packet basis without filtering*: distribute incoming packets at the ingress node to the bins in a round-robin fashion. Although it does not have to maintain any per-flow state, the method suffers from potentially having to reorder an excessive amount of packets for a given flow which is undesirable for TCP applications.
- *Per-flow basis*: distribute the flows to the bins such that the loads are similar. Although per-flow traffic filtering and distribution preserves packet sequencing, this approach has to maintain a large number of states to keep track of each active flow.

- *Hash function on the IP field*(*s*): The fields can be based on the source and destination address pair, or other combinations. A typical hash function is based on a Cyclic Redundancy Check (CRC). The purpose of the hash function is to randomize the address space to prevent hot spots. Traffic can be distributed into the *N* bins by applying a modulo-N operation on the hash space. The method of *hash function* has complexity $O(n)$ (with n the number of bins). This method is also interesting because it preserves packet sequencing and then flow nature.

After the engineered traffic is distributed into the n bins, a second function maps each bin to the corresponding LSP according to the MATE algorithm. The rule for the second function is very simple. If *LSPi* is to receive twice traffic as much as *LSPj*, then *LSPi* should receive traffic from twice as many bins as *LSPj*. The value n should be

chosen so that the smallest amount of traffic that can be shifted, which is equal to *1/n* of the total incoming traffic, has a reasonable granularity.

The main goal of MATE is to avoid network congestion by adaptively balancing the load among multiple paths based on measurement and analysis of path congestion. This approach uses a constant monitoring of the links using probe packets to evaluate link properties such as packet delay and packet loss. Using these statistics the MATE algorithm is able to optimize packets repartition among multiple paths to avoid link congestion.

Fig.2.7 depicts a functional block diagram of MATE located in an ingress node of MPLS network. Incoming traffic enters into *a filtering and distribution stage* to facilitate traffic shifting among the LSPs in a way it reduces the possibilities of having packets arrive at the destination out of order. *Measurement and Analysis stage* periodically sends probe packets to estimate congestion measure on the forward LSP from ingress to egress. The congestion measure can be delay and loss rate. The *traffic engineering function* consists of two phases: a monitoring phase and a load balancing phase. In the monitoring phase, if an appreciable and persistent change in the network state is detected, transition is made to the load balancing phase. In the load balancing phase, the algorithm tries to equalize the congestion measures among the LSPs. Once the measures are equalized, the algorithm moves to the monitoring phase and the whole process repeats. The *traffic engineering stage* splits the content of the bins among LSPs, by using a technique such as the gradient method.

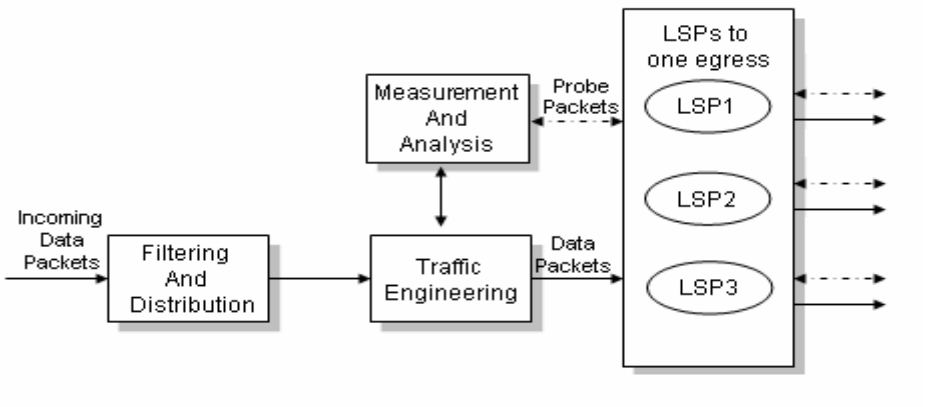


Fig.2.7 MATE Architecture from [Elwalid, 2001]

Formally a MATE network is modeled by a set L of unidirectional links. It is shared by a set S of Ingress–Egress (IE) node pairs, indexed *1, 2, 3, ..., S*. Each of these IE pairs s has a set $P_s \subseteq 2^L$ of LSPs available to it. P_s are disjoint sets. An IE pair s has a total input traffic of rate r_s and routes x_{sp} amount of it on LSP $p \in P_s$ such that

$$\sum_{p \in P_s} x_{sp} = r_{s,} \quad \text{for all } s \tag{2.12}$$

Let $x_s = (x_{sp},\ p \in P_s)$ be the rate vector of s, and let $x = (x_{sp},\ p \in P_{s,}\ s \in S)$ the vector of all rates. The flow on a link $l \in L$ has a rate that is the sum of source rates on all LSPs that traverse link l:

$$x^l = \sum_{s \in S} \sum_{p \in P_s} x_{sp} \qquad \text{for } l \in p \tag{2.13}$$

Associated with each link l is a cost $C_l(x^l)$ as a function of the link flow x^l. We assume that, for all l, $C^l(\cdot)$ is convex. Its objective is like this.

$$\min_x C(x) = \sum_l C_l(x^l) \tag{2.14}$$

$$\textit{subject to} \quad \sum_{p \in P_s} x_{sp} = r_s, \quad \textit{for all} \quad s \in S \tag{2.15}$$

$$x_{sp} \geq 0, \quad \textit{for all} \quad p \in P_s, s \in S. \tag{2.16}$$

A vector x is called a feasible rate if it satisfies (2.15–2.16). A feasible rate x is called optimal if it is a minimizer to the problem (2.14–2.16). A standard technique to solve the constrained optimization problem (2.14–2.16) is the gradient projection algorithm. In such an algorithm routing is iteratively adjusted in opposite direction of the gradient and projected onto the feasible space defined by (2.15–2.16).

The complexity of this algorithm is $O(n^3)$ where n is the number of LSPs between an ingress-egress pair of nodes. Finally, since the two stages are in sequence and if we assume that the numbers of bins and the number of LSPs are comparable, MATE complexity is in $O(n^2)$. The designers of MATE have proved in [Elwalid, 2001] that MATE's algorithm converges to an optimal routing when specific conditions are verified (see Theorem 2 page 4 in [Elwalid, 2001]).

2.5.3. Multipath-AIMD (Additive Increase Multiplicative Decrease) [Wang, 2002]

2.5.3.1. Step1 : Candidate Paths Selection

In the multipath-AIMD paper, the authors present an algorithm based on the notion of primary path. A primary path is a preferred path associated to each source. The data will then be sent mainly on the primary path, but can also use secondary paths when the bandwidth of the primary path is not sufficient.

The selection of paths in multipath-AIMD consists in selecting n LSPs equal to the number of current sources. This can be done by sorting the LSPs using their metric and then extracting the better paths. For a set of n sources, the complexity of the treatment is in $O(n\ ln(n))$. There is no stability issue in this step of the routing procedure.

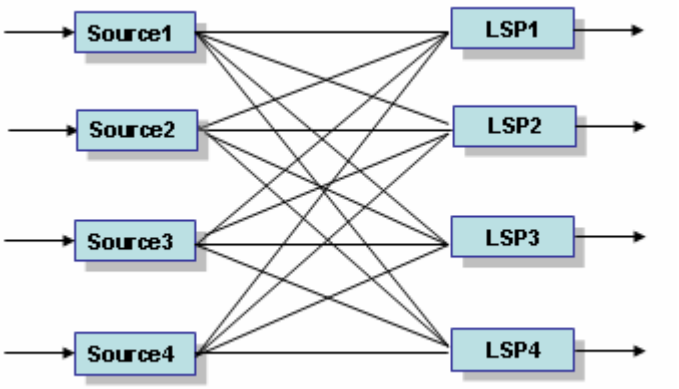


Fig.2.8 Simplified network model. LSP i is the primary path for source i [Wang, 2002]

2.5.3.2. Step2 : Distributing traffic over multiple paths

The traffic splitting uses an Additive Increase Multiplicative Decrease: starting on an average repartition, the iterative process increases the data to be sent to a path with a constant value if the link is not saturated (additive increase) and divides the amount of data to be sent to a path by a certain value if the path is saturated (multiplicative decrease). AIMD feedback algorithms are used extensively for flow and congestion control in computer networks and widely held to be both efficient and fair in allocating traffic to network paths. This approach is done in $O(n)$ complexity for n chosen paths.

In [Wang, 2002] the authors also present a modified AIMD algorithm that is closer to PPF (Primary Path First)-optimality. This solution, called multipath-AIMD with PPF correction which augments the basic algorithm to reduce the volume of secondary path traffic, is better in the way it comes closer to the expectations of the authors in terms of behavior, but it is also more expensive in resources. Its complexity is in $O(n^2)$ for n chosen paths.

Multipath-AIMD with PPF correction consists of a feedback mechanism and a rate adjustment mechanism. It takes binary feedback $f_i = \{0, 1\}$ from all LSPs. Extra feedback information is required to allow the sources to coordinate in attempting to reduce the total volume of secondary traffic. Each source $i = 1,...., N$ periodically sends messages to all other sources containing a binary vector $m_i = (m_{i1}, ..., m_{iN})$, where $m_{ij} = 1$ if source i is currently sending traffic to LSP j, and $m_{ij} = 0$ otherwise. Each source i retains the routing vector m_j associated with all other sources and each source transmits its routing vector at regular intervals of length $\Delta_{PPF} > 0$, asynchronously with respect to all other sources.

The rate adjustment mechanism for multipath-AIMD with PPF correction includes all of the rate adjustment from the basic scheme plus extra adjustments based on routing vector information received from the other sources. In particular, after each basic multipath-AIMD rate adjustment, each source in a PPF correction steps as follows.

$$x_{ij} \leftarrow \begin{Bmatrix} \max\{x_{ij} - K, 0\} & if \sum_{l \neq i} m_{li} > 0, \\ x_{ij} & otherwise, \end{Bmatrix} \tag{2.17}$$

$$x_{ii} \leftarrow x_{ii} + \sum_{j \neq i} \min\{K, x_{ij}\}, \tag{2.18}$$

Where $K>0$ is the additive PPF correction parameter. Thus if source i is making use of the secondary LSP $j \neq i$ and if LSP i is receiving secondary flow from some other source, then source i will reduce traffic on the secondary LSP j by K and, at the same time, increase its traffic to the primary LSP i by K.

From the perspective of Internet traffic engineering, multipath-AIMD seems to provide a practical mechanism for improving the utilization of LSP resources, while maintaining fairness and minimizing the complexity associated with multipath routing.

2.5.4. LDM (Load Distribution in MPLS network) [Song, 2003]

2.5.4.1. Step1 : Candidate Paths Selection

LDM limits the maximum number of extra hops to be δ to reduce the maximum amount of extra network resources as well as the number of LSPs to be set up. It is assumed that the impact of using an extra hop is relative to the scale of a network, and the scale of a network can be represented by a network diameter D. D is a maximum hop-count value of all shortest paths in a network. δ is determined in terms of D as the following:

$$\delta = D \times r \; ; \; 0 \leq r \leq 1 \tag{2.19}$$

Whenever an ingress node is initialized or a network topology change is detected, LDM computes all the paths that satisfy the above limitation, and set up LSPs for those paths. Depending on the dynamic network status, LDM selects a subset of the LSPs (candidate path set) for an ingress-egress pair, and distributes traffic load among those LSPs.

Let L_{ij} denotes the set of all LSPs set up between an ingress node i and an egress node j, and let A_{ij} the corresponding candidate LSPs, then $A_{ij} \subseteq L_{ij}$. Initially, A_{ij} is set as follows:

A_{ij} = {LSPs from i to j with the smallest hop count and with the utilization rate lower than η_0}

The utilization of an LSP, $u(l)$, is defined as the maximum of the utilization value of the links along the *LSP l*, and let $h(l)$ denotes the hop count of *LSP l*. The *utilization rate of a candidate paths set* A_{ij} is defined as the following:

$$U(A_{ij}) = min[u(l), \; \forall l \in A_{ij}] \tag{2.20}$$

LDM decides whether to expand the candidate LSP set based on the congestion level of candidate paths set. If $U(A_{ij}) \geq \rho$, then LDM further expands A_{ij}. The expansion of A_{ij}

continues, considering LSPs in L_{ij} in the increasing order of hop count until $U(A_{ij}) < \rho$ or there is no LSP left in L_{ij} for further consideration.

Generally, an *LSP* $l \in L_{ij}$ with $h(l) = (h(\textit{shortest LSP}) + m)$ should satisfy the following two conditions to be eligible for A_{ij} :

1. $u(l) < max[u(k), \ \forall k \in A_{ij}]$
2. $u(l) < \eta_m$, *where* $\eta_m < \eta_n$ *for* $m > n$

The first condition implies that LDM utilizes the LSPs with more extra hops if they have lower utilization than the LSP that has the highest utilization among the LSPs in the current A_{ij}. Moreover, since the second condition keeps the links with utilization of η_m or higher from being included in the LSPs with the path length $(h(\textit{shortest LSP}) + m)$ or longer, and $\eta_m < \eta_n$ *for* $m > n$, LSPs with more extra hops are applied with more strict utilization restriction when they are considered for the candidate LSP set eligibility. The second condition actually implies a multi-level trunk reservation. That is, links with the utilization higher than η_m can only be used by the LSPs with less than m extra hops. Note that with the candidate LSP set expansion condition i.e., $U(A_{ij}) \geq \rho$ and the first of the two eligibility conditions given above, a longer LSP is added to the candidate LSP set if the utilization of the current candidate LSP set is higher than ρ and the utilization of this longer LSP is relatively lower than the LSP with the maximum utilization among the LSPs in the current candidate paths set. With these two conditions, longer detour using an LSP with more extra hops is allowed even when that longer LSP's absolute level of congestion is already significant. The second eligibility condition enables to prevent this.

The candidate path set could either be pre-computed when there are some significant changes in the dynamic network status or be computed on demand for a new arriving user flow request. This is done in a $O(n^2)$ time in the worst case, the number of iterations growing with the utilization of the network. Here n refers to the number of paths available for the considered ingress-egress pair of nodes. In terms of convergence, the number of iterations has an upper limit defined by a given parameter δ, so the number of iterations is bounded.

2.5.4.2. Step2 : Distributing traffic over multiple paths

The traffic splitting is done using a heuristic to determine a repartition policy for incoming flows at an ingress edge router. For each incoming traffic flow, LDM randomly selects an LSP from the candidate LSP set according to the probability distribution that is a function of both the length and the utilization of the LSP. Let P_l denote the probability that LSP l is selected for sending current demand, and let us define several additional notations to explain how to compute P_l. For a candidate LSP set A, let $A = \{l_1, l_2, \ldots l_{Na}\}$, where N_a is the number of LSPs in A. C_0 is the constant to make the sum of the probabilities that are inversely proportionate to the hop count of an LSP. That is,

$$C_0 = \frac{1}{\sum_{i=1}^{N_a} \frac{1}{h(l_i)}} \tag{2.21}$$

Let $E = \max[u(sp), \forall \text{sp} \in \text{shortest LSPs in A}]$ and $d(l_i) = E - u(l_i)$, for $1 \le i \le N_a$. Then C_1 is the variable to make the sum of the ratios that are proportionate to $d(l_i)$. That is,

$$C_1 = \sum_{i=1}^{N_a} d(l_i) \tag{2.22}$$

The a_0 and a_1 factors are to be defined by the administrator to fit its needs. P_l is defined as the following,

$$P(l_i) = a_0 \frac{C_0}{h(l_i)} + a_1 \frac{d(l_i)}{C_1} \quad with \quad a_0 + a_1 = 1 \tag{2.23}$$

The route selection for a user flow takes tradeoffs between the utilization and the length of the available paths in order to enhance the performance as well as to conserve resources. As a link is more heavily utilized, it is restricted to be utilized by LSPs with less number of extra hops.

For example, if $u(l_i)$ is more than E, the shortest path is selected but $u(l_i)$ is less than E, the selection depends on the $d(l_i)$ and its length $h(l_i)$. The administrator can regulate the weight of two values with a_0 and a_1 factors.

Once probabilities are defined, each incoming flow is directed to the route with the highest probability.

The complexity of the whole splitting procedure is clearly $O(n)$. Here n refers to the number of paths selected at the end of the previous step.

LDM algorithm is very interesting because it uses two criteria simultaneously (hop count and link utilization) and it also has a low complexity. However the real problem of stability in LDM can be caused by oscillations due to candidate path selection. This oscillation problem of path selection algorithm will be explained and improved using two thresholds and it will be more developed through the hybrid schemes in the next chapter.

2.6. Conclusion

In this chapter, a functional framework of multipath routing algorithm that allows comparing different contributions for load balancing in IP based MPLS network is summarized. Most of multipath routing algorithms can be divided into two steps: *candidate paths selection and traffic splitting among them.*

This chapter also has tried to show that the link utilization ratio is the best criterion to guarantee an average QoS since this allows reducing the network congestions, packet delay and so on. Following this assessment, four recent multipath algorithms based on MPLS have been studied; WDP, MATE, multipath-AIMD and LDM, with regard to scalability and stability. The results can be summarized in Table 2.1.

Each multipath routing algorithm in the literature was declared very efficient by its authors but generally with respect to restricted conditions. In this context, instead of developing a new routing algorithm, the hybrid approach is proposed because some algorithms could be more efficient when one step combines with another step of different algorithm in the point of traffic engineering, for example by using a WDP selection of paths and a multipath-AIMD optimized PPF approach on a very powerful system, or the simple path selection of multipath-AIMD with the traffic splitting of LDM.

Table.2.1 Complexity comparison of the 4 algorithms' two stages

Algorithm	Candidate paths computation		Traffic splitting	
	Author criteria	Complexity	Author criteria	Complexity
WDP+ebp	Hop count, Residual bandwidth	$O(n^2)$	Blocking probability	$O(n)$
MATE	No selection step	-	Delay and Packet loss	$O(n^2)$
Multipath-AIMD	No selection step Need to sort	$O(n\ log(n))$	Binary feedback value of congestion	$O(n^2)$
LDM	Hop count Link utilization	$O(n^2)$	Link utilization	$O(n)$

Since the optimization objective of this study is to minimize the maximum link utilization in the network, the algorithms which use link utilization rate as a metric are selected for our hybrid approaches in the next chapter. WDP and LDM use the link utilization rate as a metric while MATE uses packet delay and packet loss. Multipath-AIMD uses binary feedback value of congestion. Therefore, especially LDM and WDP algorithm are selected for the hybrid approach in the next chapter. Besides, new traffic splitting scheme called PER (Prediction of Effective Repartition) will be proposed to place in the second step of hybrid approaches. Finally three hybrid algorithms using these modular algorithms will be proposed.

This perspective must be verified by simulation, and notably the capacity to adapt dynamically the multipath algorithm depends on the network functioning point. For an optimal bound for performance estimation, traffic bifurcation formulation is introduced. This formulation will be solved with MATLAB [MATLAB] and CPLEX [CPLEX].

Three hybrid algorithms will be simulated by *ns-2* [NS2] network simulator and *MNS 2.0* [MNS] the extension of *ns-2* for MPLS network. Finally their results will be analyzed.

Chapter 3:

Proposition of new hybrid multipath algorithms

3.1. Introduction

Our objective of Load balancing in TE is to minimize the maximum link utilization over the network. When the maximum link utilization minimizes, naturally the rejection rate for QoS demand can be reduced. If we assume that traffic grows in proportion to the current traffic pattern, this objective will ensure that the extra traffic causes minimum congestion. This chapter focuses on flow-based routing which is better than packet-based routing in the point of preventing the packet reordering overhead as discussed in the previous chapter. But this assumption implies that we know the requested bandwidth of each demand that must be routed by the network. This assumption is then only possible in case of an agreement between each user on the ISP that routes its flows.

Many authors propose different algorithms to deal with the problem of load balancing to ensure QoS requirements. Multipath routing algorithms' performances are different according to each of the two steps of the generic framework that we have explained in the previous chapter. This chapter first evaluates relevant propositions with regard to two steps that characterize multipath routing algorithm and describes some modular steps for hybrid combination.

The idea of this chapter is to combine the two steps of some algorithms to construct more performing hybrid algorithms. Since the optimization objective of this study is to minimize the maximum link utilization over the network, the algorithms which use link utilization rate as a metric, LDM and WDP, are selected for our hybrid approaches. We also propose here a new traffic splitting algorithm based on an improvement of the

corresponding part of LDM. Finally, we construct new hybrid algorithms by combining each modular steps and evaluates three hybrid algorithms. The word "*hybrid*" is used as the meaning of "c*ombination"*. In order to evaluate them, these new hybrid algorithms will be simulated using *ns-2* network simulator and *MNS 2.0* and then their results analysed. As an optimal bound for a comparison, Traffic Bifurcation Formulation [Wang, 1999] [Lee, 2002] will be solved with MATLAB.

3.2. Modular algorithms for hybrid approach

For the prerequisite of hybrid approach, every modular algorithm is presented in this part. Algorithms' basic procedures have already presented in the previous chapter. In this chapter we introduce some modifications that we propose to improve some of them. All modular parts for hybrid approaches are as follows.

- Candidate path selection part
  - Widest disjoint paths
  - LDM's first step
- Traffic splitting part
  - LDM's second step
  - New proposition called PER (Prediction of Effective Repartition)

3.2.1. Candidate paths selection - Widest Disjoint Path [Srihari, 2001]

Basic WDP path selection procedure has been explained in Chapter 2.5.1. There is one modification here. In WDP paper, width for a path set *R* is calculated as follows.

1. PROCEDURE **WIDTH(R)**
2. **W = 0**
3. While **R** ≠ ∅
4. $w^* = max\ (w_r),\ r \in R$
5. $R^* = \{r : r \in R,\ w_r = w^*\}$
6. $d^* = min\ (d_r),\ r \in R^*$
7. $r^* = \{r : r \in R^*,\ d_r = d^*\}$
8. $W = W + w^*$

9. For each l in r^*
10. $cl = cl - w^*$
11. $R = R \setminus r^*$
12. Return **W**
13. END PROCEDURE

This original procedure of WDP can be improved. WDP paper does not consider the element number of path set r^*. If the number of paths in r^* is greater than one, the line 8 is restrictive. Let us called $card(r^*)$ the cardinality of set r^*. To improve this procedure we propose to modify this line as it follows.

$$W = W + card(r^*) \cdot w^* \tag{3.1}$$

3.2.2. Candidate paths selection - Paths selection part of LDM

Basic procedure of LDM algorithm has been presented in chapter 2.5.4. The candidate path set by LDM algorithm could either be pre-computed when there are some significant changes in the dynamic network status or be computed on demand for a new arriving user flow request.

The path selection method of LDM uses time-based triggering for link state update. For a period, they use same candidates and same link information. LDM can select paths having shared links as candidates unlike WDP's disjointed paths, so the selected paths by LDM have greater congestion probability than those by WDP. This is due to the local view of LDM between two link state updates.

Since different paths between a pair of ingress-egress edge router cannot share links, WDP may be too restrictive, resulting in paths of excessive delay or in a small number of calculated paths. Both of these consequences reduce the probability that end-hosts can obtain low delay paths to their destinations. But in the point of load balancing, candidate paths are efficient if they are disjoint.

LDM algorithm for selecting candidate paths converges necessary to a solution in a finite delay because of the limitation of the number of extra-hops that are admissible to augment the number of selected paths. In the path selection algorithm given in [Song, 2003], this is expressed by the condition "$m > \delta$", m is increasing number of permitting extra hop each iteration and δ is the maximum number of extra hops to reduce the

maximum amount of extra network resources as well as the number of LSPs to be set up, that allows stopping the first loop. However the real problem of stability that can have LDM can be caused by oscillations due to candidate path selection. Each time there are changes in path utilization values the whole algorithm is applied without taking account of previous selections. Let us assume first that $\eta(t) < \rho$, ρ is utilization rate to restart to calculate new candidate paths set. Let us also assume that the load of the network becomes too important and $U(A_{ij})$, the minimum utilization of LSPs from i to j with the smallest hop count and with the utilization rate lower than η_0, becomes superior to ρ. The computation at $t+\Delta T$ will give more candidate paths than at time t. Consequently the load will be distributed on new paths (with length superior to shortest paths) implying the decrease of the utilization of each path. If the load of shortest paths decreases under $\eta(t)$, the set of computed candidate paths will come back to the situation of the time t and so on. We propose to improve the path selection algorithm by using two thresholds. The first threshold ρ_1 allows adding new paths in candidate sets. The second threshold ρ_2 must be lower than the first one. If the minimum of candidate path utilization go under this threshold, this enables to reset the candidate paths selection by restarting the whole algorithm (Fig 3.1). Original LDM's candidate path selection procedure is in Annex B.

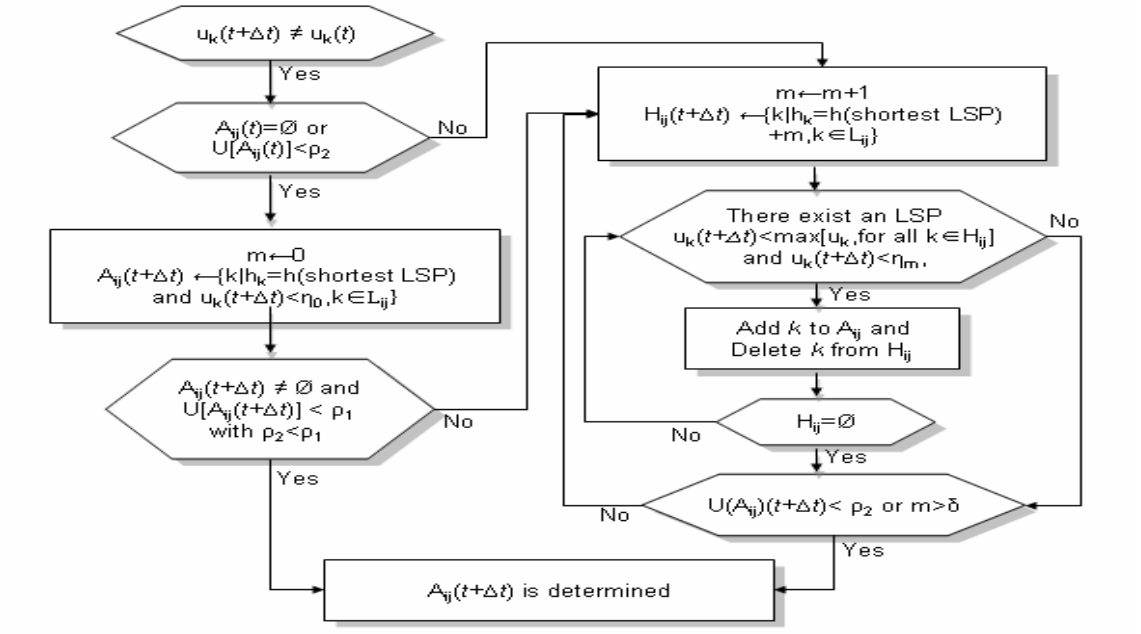


Fig. 3.1 Modified LDM candidate path selection algorithm [Lee, 2006b]

3.2.3. Traffic splitting - Traffic splitting part of LDM

The traffic splitting is done using a heuristic to determine a repartition policy for incoming flows at an ingress edge router. For each incoming traffic flow, LDM randomly selects an LSP from the candidate LSP set according to the probability distribution that is a function of both the length and the utilization rate of the LSP.

3.2.4. Traffic splitting - PER (Prediction of Effective Repartition)

We propose a new traffic splitting algorithm based on an improvement of the corresponding part of LDM.

QoS routing algorithms look for the optimal path between source-destination nodes pair based on the network state information obtained from the network state database. New proposition's main idea in MPLS network is that nodes do not select paths based on their network state information, but also on predicted information until next link state update time. The predicted path is obtained from some predicted information kept in the resource manager.

New proposition calculates each probability on candidate paths like LDM but it is different from LDM in the point of how it selects one path to assign the current flow. It uses the differences between its computed probability and the effective utilization rate after loading the traffic on it. The state after effective repartition is measured using the predicted information kept in the traffic engineering manager and compared to the pre-computed probability. This information of difference between both states is used to take a decision about the action to take to reduce that difference.

The periodic update policy is chosen in this thesis to provide partial link state information in the network. The periodic policy is chosen because of its simplicity. An estimation and forecast algorithm is used to compensate for the lack of accurate information. This new algorithm is called PER (Prediction of Effective Repartition) algorithm because it uses the predicted information of effective repartition to make a decision.

3.2.4.1. Calculating a repartition probability

Suppose that traffic splitting algorithms maintain packet ordering within a flow. Since we cannot split individual flows, consecutive packets from the same flow can be viewed as a single logical packet train that cannot be further divided by the traffic splitter. Thus a large number of long packet trains will have negative effects on the traffic splitting performance. Traffic splitting is significantly harder when there is small number of large flows. The traffic splitting is done using a heuristic to determine a repartition policy for incoming flows. Each path is adjoined a repartition probability using the following formula (3.2) (3.3) and (3.4). The calculation of the probability uses the traffic matrix which is made in the last link state update time.

For each incoming traffic flow, ingress router selects one LSP among the candidate paths according to two probabilities: calculated repartition probability r_i and effective distributed ratio between two update times e_i. Using these two values, we calculate new value S_i for selecting one LSP to assign among candidate paths.

We define several notations to calculate r_i, e_i and S_i. Suppose that candidate path set is $CP = \{l_1, l_2, ..., l_k\}$ where k is the number of the selected paths.

In equation (3.2), H is the constant to make the sum of the probabilities that are inversely proportionate to the hop count of an LSP. This value is the same as the value C_0 of LDM's splitting algorithm.

$$H = \frac{1}{\sum_{i=1}^{k} \frac{1}{hc(l_i)}} \tag{3.2}$$

Let $b(i)$ is the bottleneck's residual bandwidth of LSP_i in CP. B is the sum of $b(i)$.

$$B = \sum_{i=1}^{k} b(i) \tag{3.3}$$

r_i is the function of both the length and the residual capacity of LSP 'i' and it represents its splitting probability. r_i is given by equation (3.4), with p_0 and p_1 factors defined by the administrator to fit its needs.

$$r_i = p_0 \frac{H}{h(i)} \quad + \quad p_1 \frac{b(i)}{B} \quad with \quad p_0 + p_1 = 1 \tag{3.4}$$

e_i is the effective splitting ratio on *LSP* '*i*' in the candidate paths. It is calculated as the ratio between the current traffic on the total demands in the traffic class from last link state update time until now.

$$e_i = \frac{\sum_{j=1}^{n_i} d_{ji}}{\sum_{j=1}^{n} d_j} \quad where \quad \sum_{i=1}^{k}\sum_{j=1}^{n_i} d_{ji} = \sum_{j=1}^{n} d_j \tag{3.5}$$

- n_i is the number of flow demands assigned to LSP '*i*'
- n is the total number of demand, i.e, $n = \sum_{i=1}^{k} n_i$
- k is the number of candidate LSPs
- d_{ji} is the traffic amount of the ji th demand assigned on LSP_i
- $0 \le e_i \le 1$

3.2.4.2. Selecting one LSP with bandwidth constraint

S_i is the value to estimate how much of r_i is satisfied with its repartition probability in real. The prospected state e_i is measured and compared to the desired state r_i. This information or difference between both states S_i is used to take a decision about the action to reduce that difference. Ingress router calculates the differences between its pre-computed probability r_i and the effective repartition ratio e_i. Then the probability of selection S_i for each candidate path is calculated by the equation (3.6).

$$S_i = \frac{r_i - e_i}{r_i} \tag{3.6}$$

- $S_i > 0$

If S_i is greater than 0, it means that effective repartition is less than the calculated objective. The bigger S_i is, the bigger is the distance to reach the objective. In this case the control consists in selecting the LSP_i with the biggest S_i.

- $S_i \leq 0$

If S_i is less than 0, it means that effective repartition exceeds the repartition objective of the period. In this case, there is a risk of link congestion of the path. So the algorithm must prevent of sending current traffic on this path. If all paths in the candidate path set have a negative S_i, it restarts a new candidate path selection procedure.

Let us recall here that we assume that we know the requested bandwidth of each incoming demand. Since the residual bandwidth *b(i)* of each LSP is different and because the link state update of each ingress router is performed periodically and asynchronously with regard the other ingress routers, each ingress node must control that the cumulated amount of the demand that it assigned to one of its LSPs will not provoke a congestion. This implies that even when an LSP has the greatest parameter S_i, it is necessary to verify that it has sufficient capacity to transport the current demand.

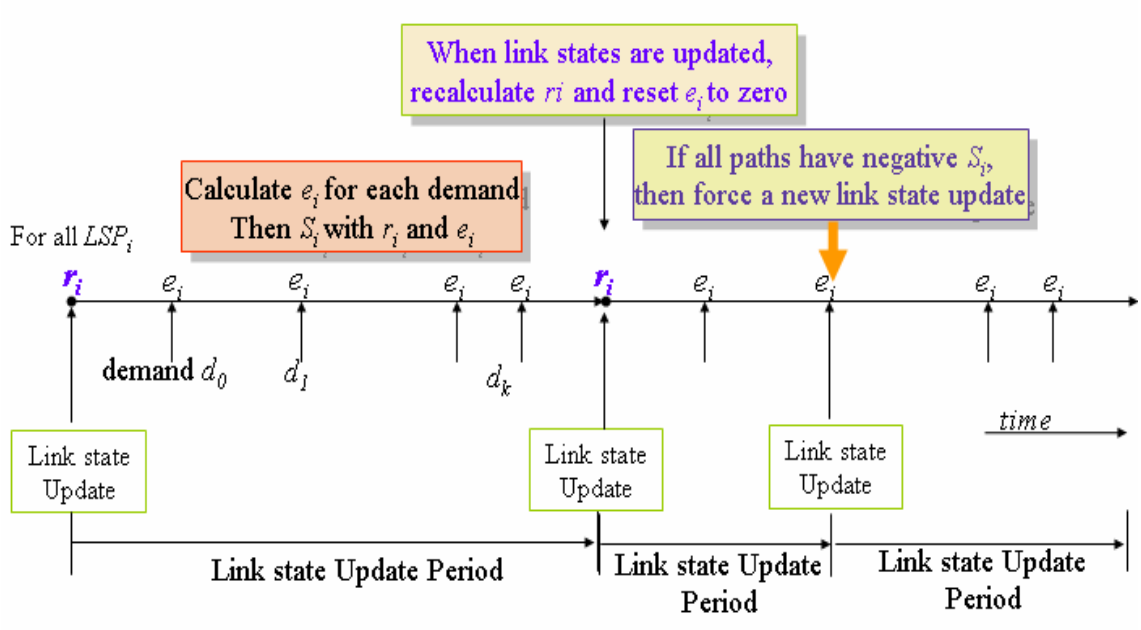


Fig.3.2 PER parameters

As illustrated by Fig.3.2, PER traffic splitting has a periodic functioning. At the beginning of each period, a link state update fills the ingress router database with

current values of the parameter such as the residual bandwidth of each LSP managed by the router. With these values, it calculates for each LSP_i its splitting objective rate r_i for the current period. When a new demand arrives, it calculates the effective splitting rate e_i and the selection parameter S_i of each LSP_i. Thus the demand flow is assigned to the LSP with the greatest S_i that owns enough residual bandwidth. After the ingress router updates the residual bandwidth of the assigned LSP by removing the quantity corresponding to the bandwidth requested by the demand. When all its LSPs have a negative selection parameter S_i, the ingress router forces a new update to define a new routing map based on a new candidate set of LSPs and accurate network values with regard to assignment achieved by the others ingress routers.

- Notations used by PER algorithm.

  For *LSP i*

  - *b(i)* : residual bandwidth
  - r_i : splitting objective rate
  - e_i : splitting effective rate
  - S_i : selection parameter

  For demand d_j

  - $bw(d_j)$: requested bandwidth per timeout

Fig. 3.3 shows the flowchart of our PER algorithm. Since this algorithm only addresses the problem of demand splitting, in case of impossibility to process a new demand, it just forces a link state update. Further, we will see that after this forced link state update, we start the candidate path selection algorithm to re-qualify the set of candidate paths.

LDM can send all traffic onto the same path calculated by its probability value until next trigger time in the worst case because LDM selects one LSP in the candidate LSP set randomly. But, because new PER method uses its own database calculated temporarily, it can avoid congestion state better than LDM.

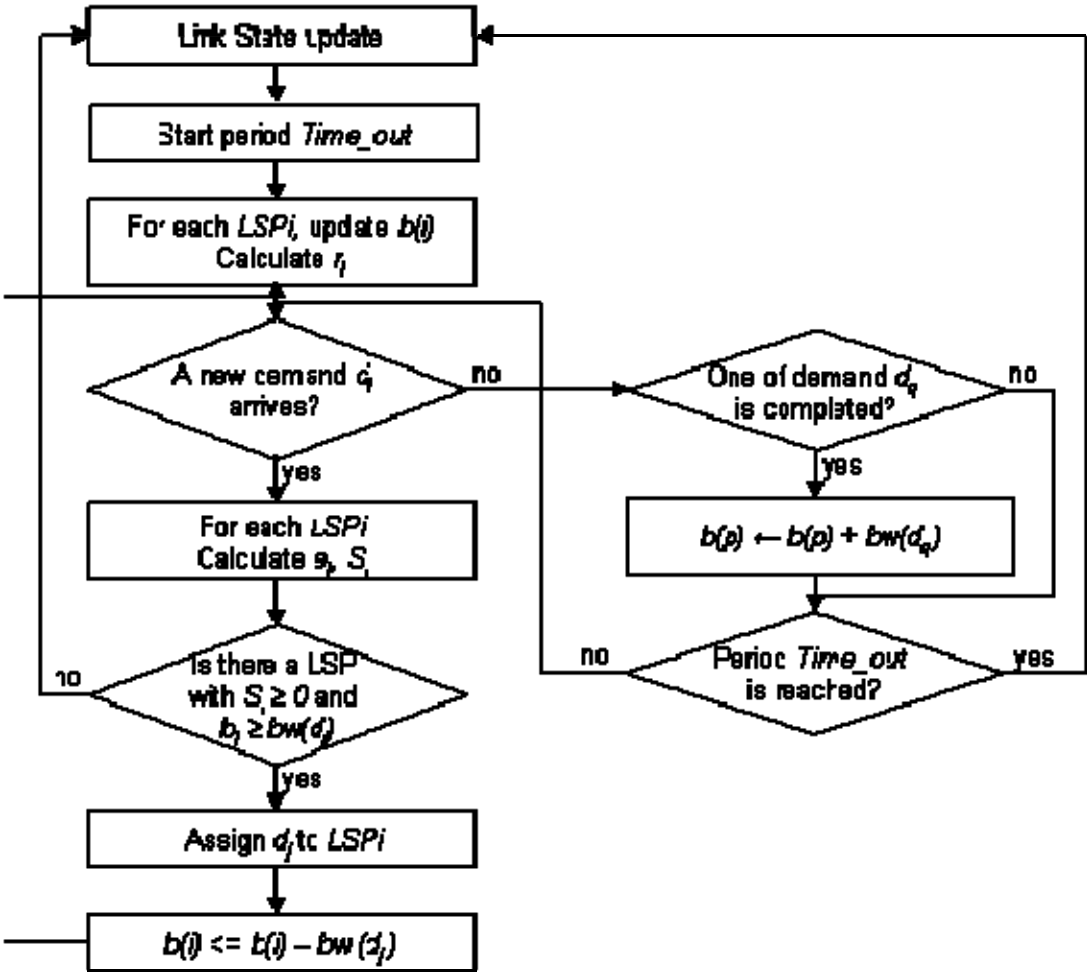


Fig. 3.3 Flowchart of PER algorithm

3.3. Proposition of three hybrid multipath algorithms [Lee, 2006a]

The QoS for a dynamic approach should be measured and bandwidth allocations adjusted accordingly. If the adjustment cannot keep up with the traffic dynamics, good QoS cannot be provided. Obviously, better QoS can be expected by more frequent bandwidth reallocations. But monitoring QoS and changing the bandwidth allocations too often in a high-speed core router may lead to dramatic computational overhead. Another problem with this dynamic approach is that connection admission control becomes difficult as the bandwidth allocations change in time.

In this context, we propose three hybrid algorithms using modular steps presented in sections2.5 and 3.2 in order to minimize the maximum link utilization over the network.

Two modular algorithms for the candidate paths selection step and two for traffic splitting step have been explained respectively. Three combinations are possible after eliminating the original LDM algorithm. Original LDM algorithm will be used in the part of performance estimation to prove whether new hybrids' performances improve original LDM's. LDM's problems are as follows.

- From the first step: its congestion probability is higher than when using disjoint paths since its candidate paths share their links each other.
- From the second step: each path's selection probability is calculated, but it is not well adapted because it selects one path randomly among candidate paths.

First and second hybrids are improvement of original LDM through exchanging one step of two to another scheme. First hybrid is to improve first step of LDM and second hybrid is to improve second step of LDM. And then last combination called LBWDP (Load Balancing over WDP algorithm) is proposed newly using two modular parts which are used in the precedent two combinations to improve original LDM.

3.3.1. Hybrid1 : WDP and LDM splitting algorithm

WDP for candidate path selection part is combined with LDM's traffic splitting part for the first combination. It is for improving the first step of original LDM algorithm.

In the algorithm given by Fig. 3.4, let us consider the following notations:

- L_{ij} are all possible paths from source i to destination j.
- CP_{ij} is the candidate path set of the current period.
- d_k is the current traffic demand.
- CP_{final} is the candidate path set that is explored to select an LSP_i to handle a new demand.
- CP_{hist} is a copy of current CP_{ij}. If CP_{final} is vacant, the algorithm recalculates new candidate paths except for the elements of this CP_{hist}.

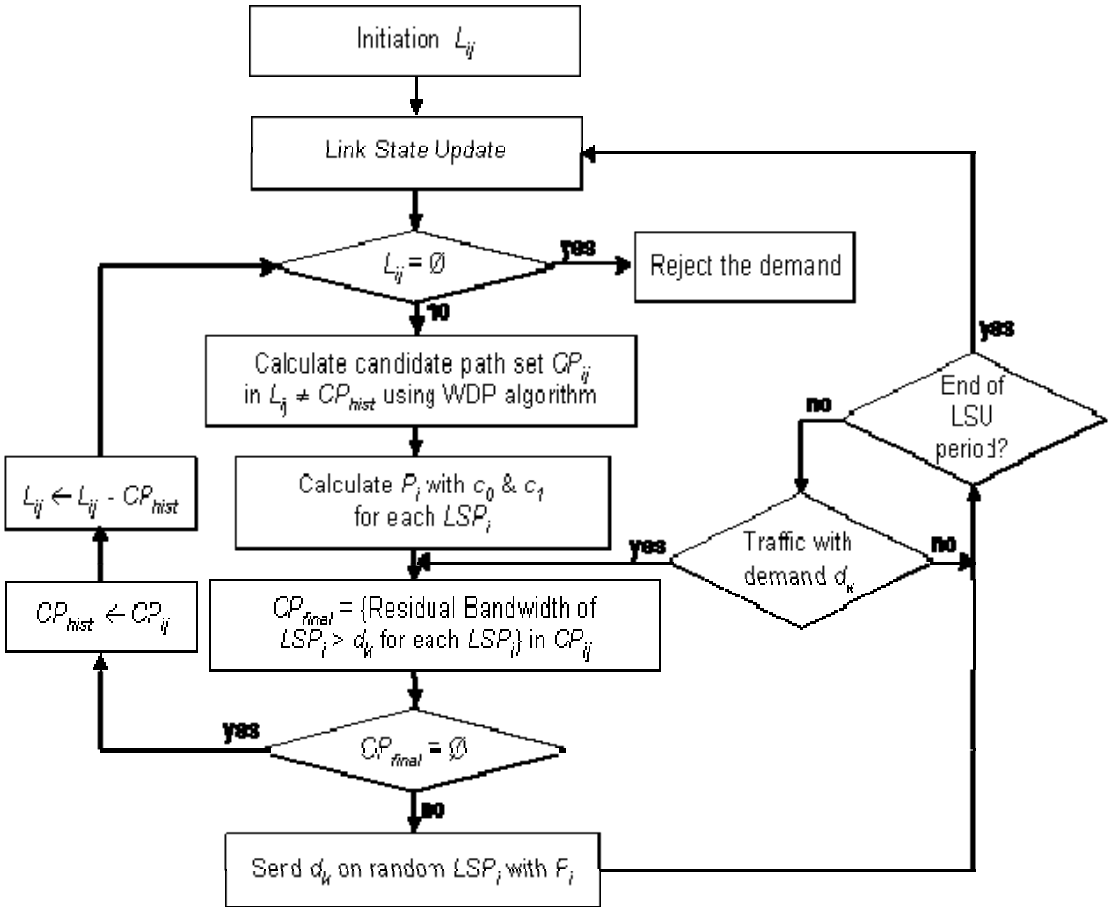


Fig. 3.4 Flowchart of Hybrid 1

To start, it calculates candidate paths (CP_{ij}) using WDP algorithm in all possible paths (L_{ij}) from source to destination nodes every link state update time. For each candidate path, its probability value (P_i) using each candidate path's hop-count and link utilization is calculated according to LDM's method. On arrival of a new demand (d_k) to send, it selects one path randomly with their probabilities among candidate paths. At that time it deletes the paths which have an insufficient capacity to assign the current traffic from the candidate paths set (CP_{final}). If all paths in the candidate path set are not sufficient, it recalculates new candidate paths set except them (CP_{hist}). On selecting one path to assign the current demand, it sends the traffic through that path. But, since LDM algorithm selects it randomly, its congestion probability can be increased and it needs better choice.

This algorithm is better with regard to congestion than original LDM, because its path independence characteristic of candidate paths. In LDM scheme, since candidate paths can share their links each other and traffic splitting scheme uses same candidate paths to select one path to send the traffic till the next link update time. Its congestion probability is higher than when disjoint. In this hybrid, by using WDP algorithm for candidate path selection, congestion probability is reduced and maximum link utilization in the network is decreased comparing to original LDM.

Nevertheless it must be notice then between two link state update, the ingress router does not update the residual bandwidth of its LSPs, in spite of its assignment of demands. This explains why even in the case of only one pair of ingress-egress nodes, it remains possibilities of congestion.

3.3.2. Hybrid2 : LDM selection algorithm and PER

The second hybrid is LDM's candidate path selection part with PER traffic splitting scheme. It is for improving the second step of original LDM algorithm. Modified LDM algorithm in the chapter 3.2.3 is used especially for this hybrid algorithm.

In the algorithm given by Fig. 3.5, let us consider the following notations:

- m is a variable to increase the hop count
- k is a LSP of L_{ij}
- h_k is the path set having less than hop count limit in the L_{ij}
- H_{ij} is a temporary candidate path set. If an LSP in H_{ij} does not surpass the link utilization limit, it becomes the element of CP_{ij}.
- $U(CP_{ij})$ is minimum utilization rate of the paths in candidate path set CP_{ij}
- ρ is the maximum admissible utilization rate for currently used paths.
- δ is the maximum extra hop-count length with regard to the shortest LSP length.

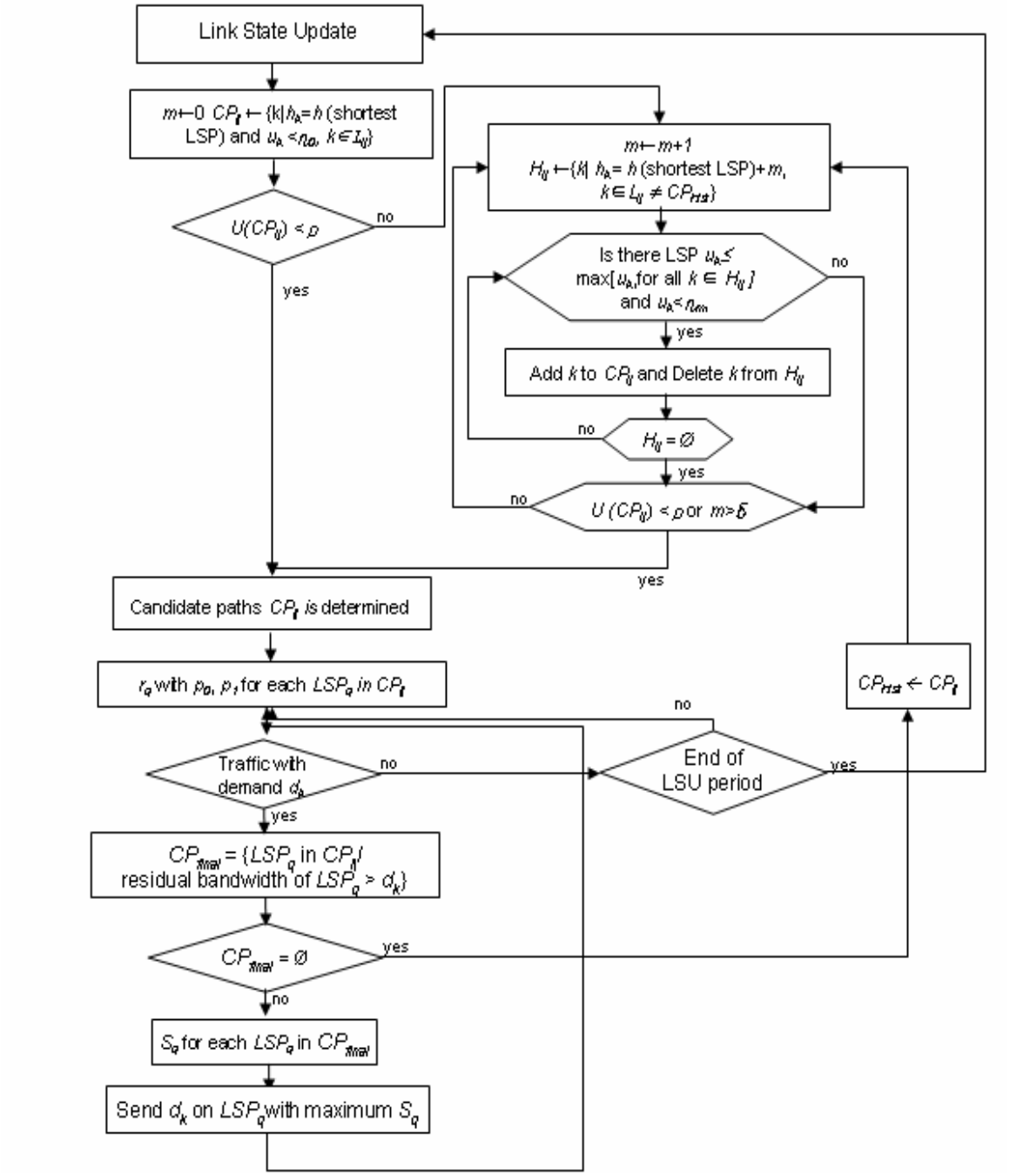


Fig 3.5 Flowchart of Hybrid 2

Candidate path set (CP_{ij}) is computed according to LDM algorithm starting with the shortest paths (h_k=h(*shortest LSP*))

The advantage of PER method is to reduce the probability of congestion and balance well the load owing to pre-calculating the effective utilization ratio of candidate paths. But this method needs much calculation.

3.3.3. Hybrid3 : LBWDP (WDP and PER) [Lee, 2005b]

First hybrid can improve its performance by putting WDP algorithm in the first step of original LDM and second hybrid can also improve by putting PER algorithm instead of LDM's second step. For candidate path selection, WDP algorithm is better than LDM in the point that WDP can reduce the congestion probability owing to links' disjointedness, and for traffic splitting, PER algorithm is more efficient than LDM by using S_i splitting parameter.

The last hybrid is thus the combination of two better modules in each step, WDP for candidate path selection and PER for traffic splitting. It is specially called LBWDP (Load Balancing over WDP algorithm). Candidate paths are computed using WDP algorithm every link state update time, the traffic is assigned onto the best path through PER algorithm when a demand is arrived. This traffic engineering scheme will be useful for reducing the probability of congestion by minimizing the utilization of the most heavily used link in the network.

This LBWDP algorithm is theoretically the best in the point of efficiency and reducing the congestion but it is worst in time and computation complexity. Figure 3.6 depicts the flowchart of LBWDP.

In the algorithm given by Fig. 3.6, let us consider the following notations:

- IBW_i is the initial residual bandwidth of LSP_q. It is the minimum of the unused bandwidth of all the links composing LSP_q.
- Between two link state update periods, BW_q represents the supposed amount of unused bandwidth of LSP_q. Since WDP only assume disjointed paths with regards to a pair of ingress-egress routers, the real value can be smaller than this expected value that represents the best case. After a link state update, BW_i is initialized with the value of IBW_i.

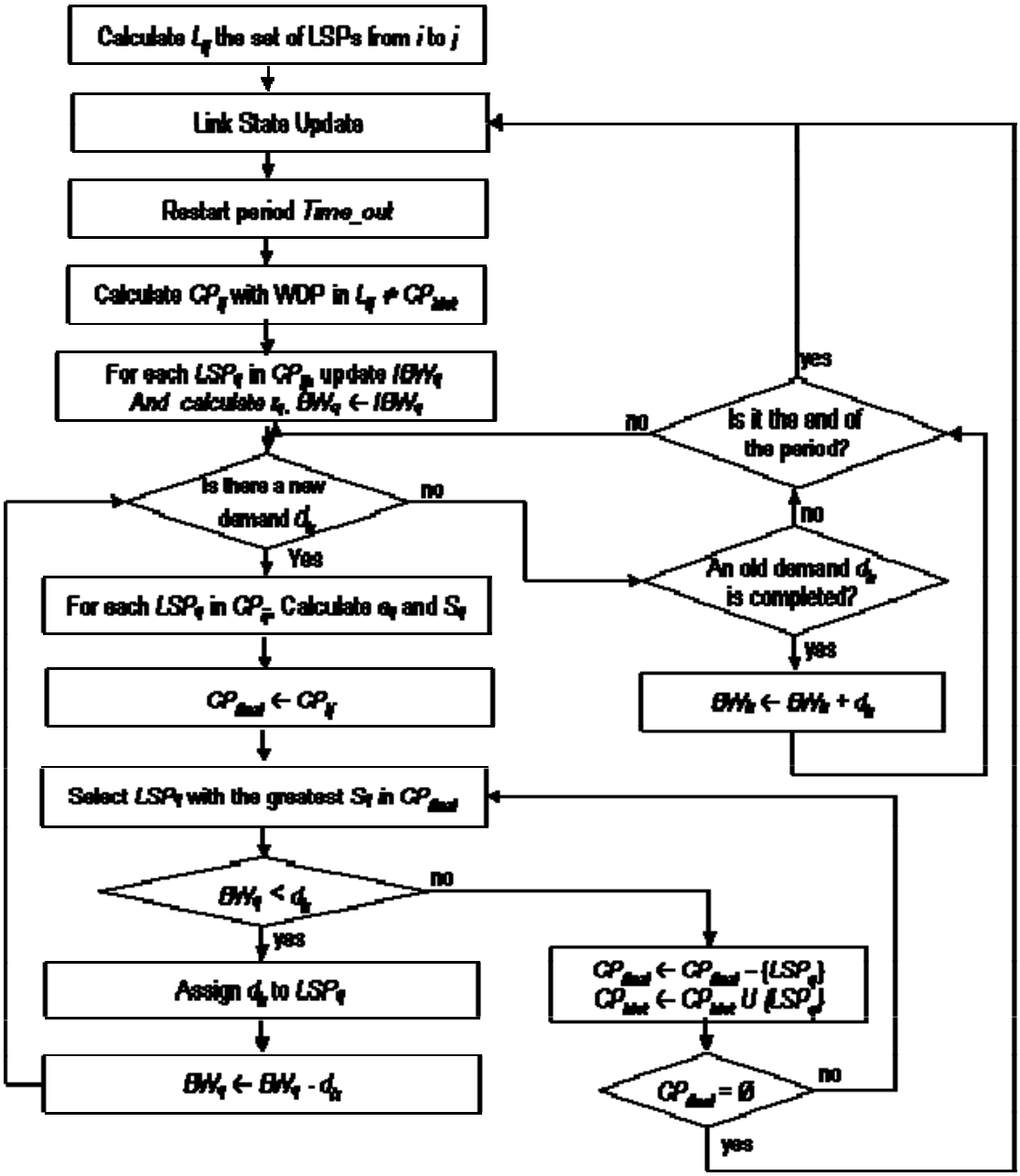


Fig 3.6 Flowchart of hybrid 3 (LBWDP)

First, it calculates candidate paths (CP_{ij}) using WDP algorithm in all possible paths (L_{ij}) from source to destination nodes every link state update time. For each candidate path, LSP 'q', its probability value (r_q) using each candidate path's hop-count and link utilization is calculated according to equation 3.4 when a new demand (d_k) arrives, it calculates the effective repartition rate (e_i) on each *LSPi* in the candidate path set (CP_{ij})

and calculates the splitting selection parameter (S_q) using the difference between desired (r_q) and effective (e_q) values. It deletes the candidate paths set (CP_{final}) which have an insufficient capacity to assign the current traffic. If all paths in the candidate path set are not sufficient, it recalculates new candidate path set except them (CP_{hist}). It selects the LSP with the greatest S_q value. On selecting one path to assign the current demand, it sends the traffic through that path.

It is concluded theoretically that the best hybrid is the combination of WDP for candidate path selection and PER for traffic repartition. WDP can reduce congestion probability and balance the load owing to its disjointedness comparing LDM's path selection procedure and PER owing to its prediction better than LDM's splitting procedure. The simulation result will prove that.

3.4. Performance Comparison

In this subsection, the proposed hybrid algorithms are compared with other well-known algorithms from the viewpoint of different metric as the maximum utilization rate of network's links.

Given users' demands, the sources calculate their paths of traffic flows in order to minimize the maximum link utilization. For a traffic demand, a flow is allocated on a single path.

Generally there are two forms of network protocol evaluation, analytical modelling and computer simulation. The first is by mathematical analysis that characterizes a network as a set of equations. The main disadvantage is it's over simplistic view of the network and inability to simulate the dynamic nature of a network. Thus the study of complex design system always requires a discrete event simulation package, which can compute the time that would be associated with real events in real life situation.

In this paper, two methods are used for performance comparison. To get an optimal value of traffic engineering, we solved the traffic bifurcation LP (Linear Formulation) problem with MATLAB using the language Visual C++. The values given by the LP

solver are compared with our three hybrid algorithms' results given by simulation. They have been obtained with the following simulators.

- *ns-2*: It is an IP based discrete event driven simulator widely accepted and used for the analysis of computer networks. It is developed at UC Berkeley and is currently maintained at USC. We have used version 2.1b8a all-in-one for Linux Redhat7.3, with the addition of the *MSN2.0* patch to simulate the MPLS features [NS2].
- *MNS 2.0*: NS2 has been integrated with the MNSv2 patch. It is an MPLS extension for *ns* implemented by [MNS2.0] which supports simple MPLS function such as LDP (Label Distribution Protocol) and label switching, as well as CR-LDP (Constraint-based LDP) for QoS routing without constructing a real MPLS network.

The simulation in this chapter is carried out with the same factors for all the experiments, so that a common ground can be reached for comparison purposes.

3.4.1. Simulation topology

The simulation topology is a MPLS-based IP network. The network is composed of twelve routers from LSR1 to LSR12 (Fig. 3.7). The bandwidth between two routers is specified on the link between them. Three sources of traffic and three destinations are connected to some of these routers. Possible source-destination pairs are Src0-Dst0, Src1-Dst1 and Src2-Dst2.

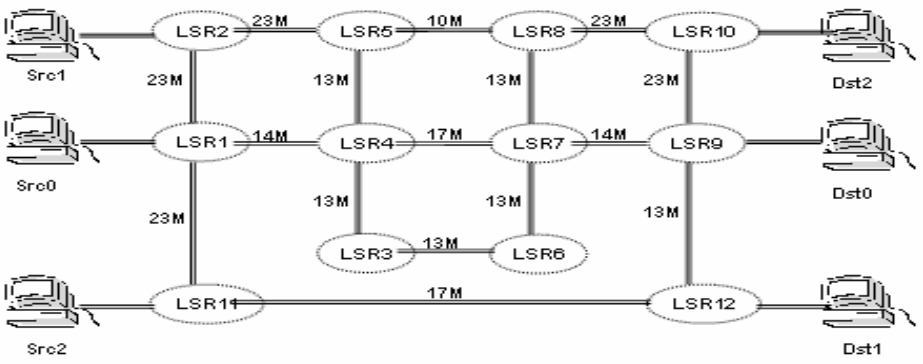


Fig. 3.7 Simulation topology

3.4.2. Simulation scenarios

Traffic scenario for multi-users is as follows.

- The volume of each individual demand is fixed at 300kb per second.
- Simulations with different number of demands are executed for 150 seconds respectively.
- Source-destination pairs are selected randomly among them every demand generation interval.
- One flow generated in certain time is stopped in a random time.
- Each demand is required randomly selected from uniform distribution
- The link state update period is fixed at 3 seconds.
- We simulate all the algorithms with same traffic scenario changing the traffic volume in the same environment.

Our proposed hybrids' simulation results given by *ns-2* are compared with the traffic bifurcation solution *TB*(*1*) given by MATLAB as an optimal bound. Let us notice that as in our simulation *TB*(*1*) does not split an incoming demand to several LSPs. We simulate also other existing routing algorithms, Shortest Path algorithm and Shortest Widest Path algorithm, and compare their results with our hybrid algorithms' such as LBWDP's.

- Shortest Path (SP hereafter) [Dijkstra, 1959]: selects the path that has minimum hop count.
- Shortest Widest Path [Wang, 1996] (SWP hereafter): selects the paths in which the minimum unreserved bandwidth of the links along the path is largest among all the paths and whose unreserved bandwidth exceeds the required bandwidth of the LSP. It selects the path with the largest residual bandwidth. If there are multiple feasible paths, the one with minimum hop count is selected.

3.4.3. Simulation results

All simulations are executed and compared changing the performance metric (maximum link utilization, packet loss rate and rejection rate).

3.4.3.1. Metric1 : Maximum Link Utilization rate

We have derived the theoretical lower bound of the maximum link utilization using TB formulation (see section 2.4.2) to evaluate the performance of the three proposed hybrid algorithms. We have imagined these combinations from the effort to improve original LDM algorithm. Our results are naturally compared with the performance of LDM whether our hybrids' performances are superior to it and we compare them with the traffic bifurcation results.

In the simulation, demand generation rate is one demand every 1 to 20 seconds. Table3.1 illustrates: the maximum link utilization of TB, SP, SWP and LBWDP as the number of demands grows. During the simulation, maximum link utilization over the network is logged.

When network load is light, all algorithms perform well. Performance difference is obvious when network load increases and traffic of some link is heavier than other links. The reason is that heavy load cause network to be congested and only some of the source and destination pair located near can reserve resource for traffic.

Table3.1 shows that the SWP is good in light network loads and starts to show performance degradation when network load increases but insignificant. LBWDP is better than SWP algorithm when network load increases. SWP algorithm is relatively efficient when there is enough capacity but it degrades remarkably when demands are too much while LBWDP make a balance with the same volume of traffic. And we observe that even though performance of LBWDP is not much efficient with enough capacity, but it is evident that it does not degrade as the number of demands grows. It is very important how efficiently can be balanced the load over the network when the demand is much more to be congested. As a result, LBWDP is more efficient than others in load balancing of the network when the demands grow more and more. We

depict these results in a graph Fig.3.7. As depicted in Fig.3.7, even when there is saturation with SP and SWP algorithms, LBWDP achieves well load-balanced network.

We can propose multi-model approach that more than two algorithms changing the load can adapted. That is, when load is light, SP or SWP algorithm is taken and when load is heavy to congest, LBWDP algorithm is used.

In general, when link utilization is greater than 1, the request will be blocked because of scarce network bandwidth assuming that over-booking is not permitted. We marked the ratio of blocked requests to the total requests in every interval which cause the link utilization to be greater than 1.

Demands	TB	LBWDP	SP	SWP
one/20sec	0.0883	0.3250	0.2890	0.3610
one/10sec	0.1060	0.4160	0.3600	0.3010
one/5sec	0.2158	0.5955	0.8471	0.4984
one /3sec	0.2681	0.4898	1.0329	0.5632
one/2sec	0.2229	0.5866	1.2832	0.7822
one/1sec	0.5405	0.6352	3.5604	1.5040

Table3.1. Simulation results: maximum link utilization

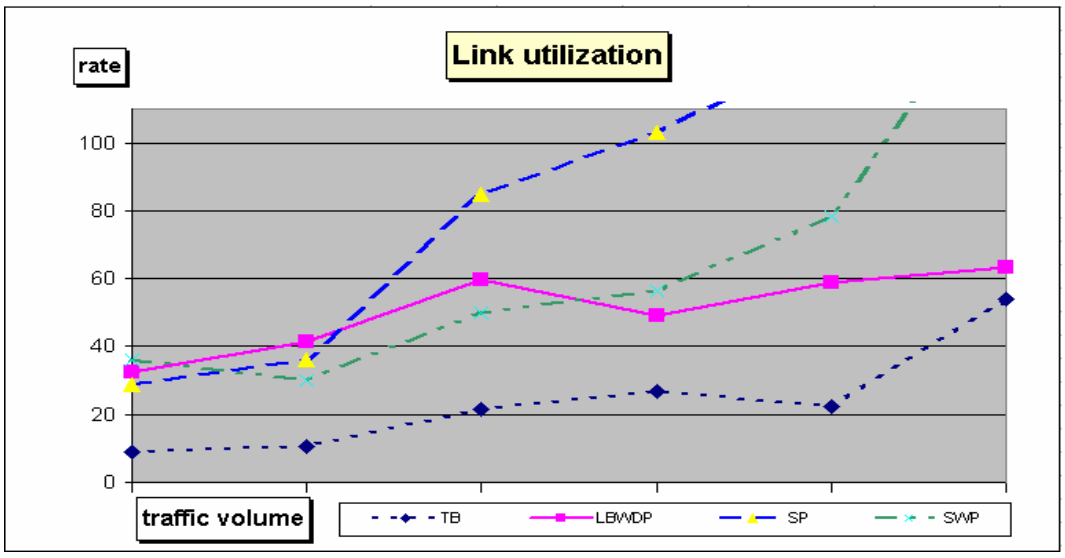


Fig.3.8. Comparison of maximum link utilization

Fig.3.8 shows well that LBWDP get the best performance that has minimum link utilization of network when the load is high. When the number of shared links between LSPs increases and when the traffic to send also increases, then the difference between the information kept and the real state increases too. When the difference between the local estimation of residual bandwidth and its real value given by next link state update is high, congestion probability is increased because of the inaccurate information. PER algorithm reduces this inaccuracy. When traffic volume is high, PER proves its efficiency.

Table3.2. represents the average link utilization of the compared schemes. As the traffic load grows larger over a certain level, LBWDP's link utilization increases very little due to the trunk reservation restriction. As the traffic load grows, the average link utilization of SWP increases most rapidly, and the link utilization of SWP has the highest value when the traffic load is the heaviest among the experimented cases. Though SP's average link utilization is lower than others excepted for the last simulation case, it is because SP continues to send the traffic through one path and lost much of them. For further traffic load increases, though, the link utilization increment slows down in LBWDP. With average link utilization, we conclude that LBWDP can make well load balanced network without using large amount of network resources and with low packet loss rate as depicted in 3.4.3.2.

Demands	**SP**	**SWP**	**LBWDP**
One/20sec	0.0647	0.0889	0.0820
One/10sec	0.0719	0.0887	0.0971
One/5sec	0.1461	0.1925	0.2009
One /3sec	0.1398	0.1817	0.1568
One/2sec	0.2343	0.2693	0.2429
One/1sec	0.3167	0.3438	0.2587

Table3.2. Simulation result: Average network utilization

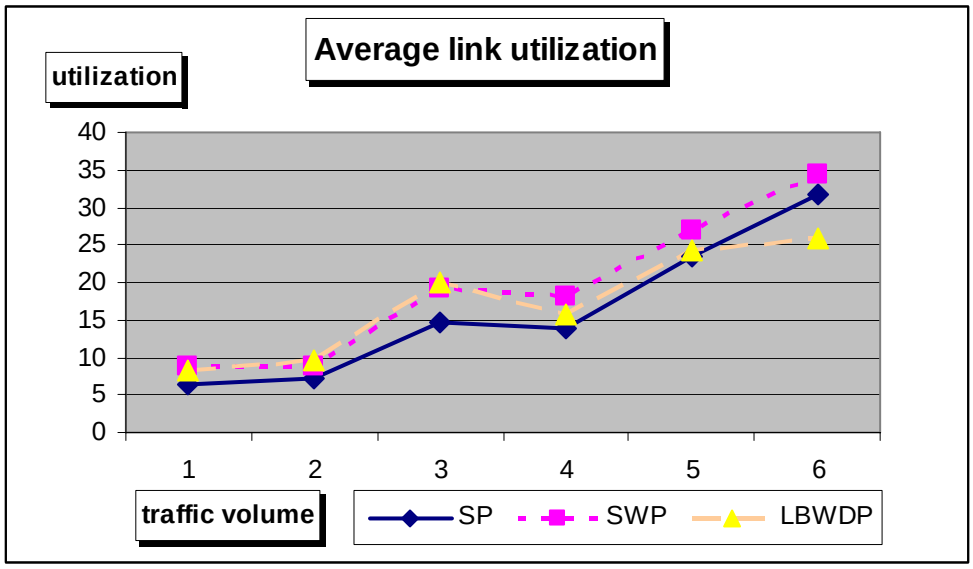


Fig.3.9 Comparison of average link utilization

Table3.3 is the simulation result about maximum link utilization for proposed three hybrid algorithms. It shows that the most efficient algorithm for load balancing is LBWDP (hybrid 3), and then hybrid1 (WDP + LDM), hybrid2 (LDM + PER).

Number of demands	LBWDP	WDP+LDM	LDM+PER
one/20sec	0.3250	0.3360	0.2890
one/10sec	0.4160	0.3590	0.3560
one/5sec	0.5955	0.5986	0.5879
one/3sec	0.4898	0.6275	0.7300
one/2sec	0.5866	0.8170	0.9384

Table3.3. Simulation result: maximum link utilization of 3 hybrids

As illustrated by Fig.3.10, LBWDP is particularly efficient when the load of the network is important.

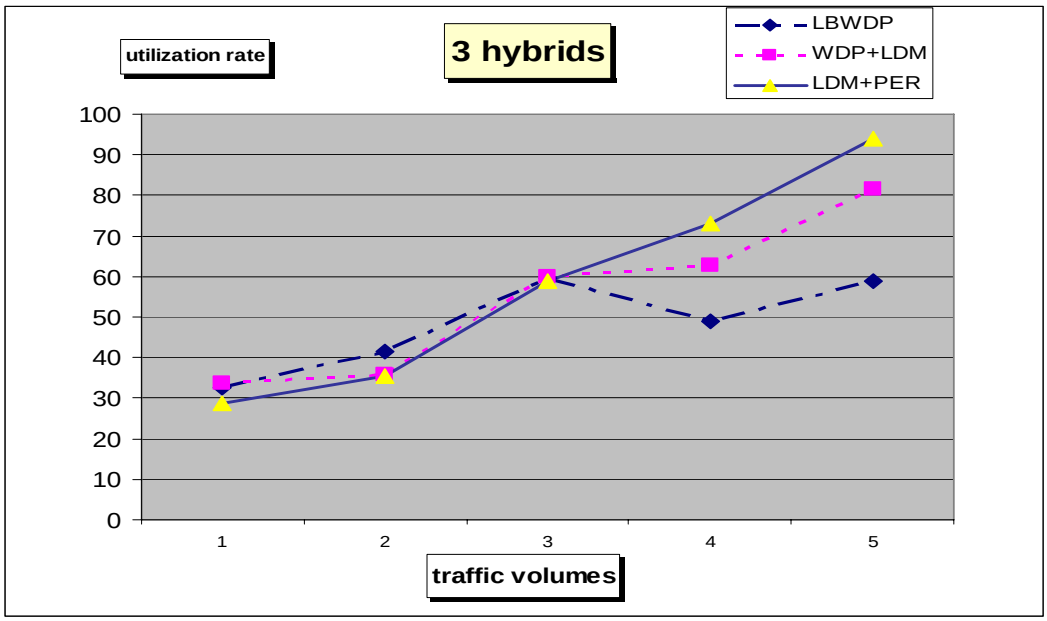


Fig.3.10 Comparison of 3 hybrids' average link utilization

3.4.3.2. Metric2 : Packet loss rate

Packet loss rate is also measured as the metric for the network performance perceived by the users.

Some applications do not perform well (or at all) if end-to-end loss between hosts is large relative to some threshold value. The sensitivity of real-time applications and of transport-layer protocols to loss become especially important when very large delay-bandwidth products must be supported [RFC2680].

Demands	SP		SWP		LBWDP	
	Recv.(lost)	Loss rate(%)	Recv.(lost)	Loss rate(%)	Recv.(lost)	Loss rate(%)
one/20sec	12472 (0)	0	12468 (0)	0	12467 (0)	0
one/10sec	14294 (0)	0	14289 (0)	0	14283 (0)	0
one/ 5sec	33736 (1345)	3.83	33030 (2032)	5.80	32157 (0)	0
one / 3sec	33239 (6615)	16.60	36731 (3116)	7.82	27190 (0)	0

	SP		SWP		LBWDP	
one/ 2sec	47350 (10493)	18.14	47626 (10238)	17.69	36640 (74)	0.20
One/ 1sec	53336 (83417)	61.00	56328 (81123)	59.02	40797 (131)	0.32

Table3.4. Simulation result: Packet loss rate

Table3.4 shows the received packet, lost packet and packet loss rate given by simulations for SP, SWP and LBWDP algorithms. As showed in Fig.3.8, two algorithms, SP and SWP, do not control the congestions. Because algorithm SP always sends the traffic on the one shortest path even though it is congested, and SWP also sends to one shortest-widest path until the next link state update time. Consequently, these two algorithms could not avoid the congestions and packet losses.

It shows that the packet loss rate of SP is much higher than others, since only the shortest path is used, limiting the total network core bandwidth that can be used. SWP also has high rate when load is heavy. But LBWDP has always low packet loss rate even when load is heaviest because of its load balancing power. This result proves that the link state utilization rate is indirectly a good criterion to improve the packet loss rate.

3.4.3.3. Metric3 : Demand Rejection rate

After demand has been generated, candidate paths for each demand are searched using routing algorithms which performance is to be checked. If the algorithm finds a path for connection, each source tried to set up path along the returned path. It is considered a rejected call when a routing algorithm can not find a feasible path. While during a demand setup, demand will be rejected when one of the links along the path from source to destination does not fulfil the requested bandwidth.

Demand rejection rate is the percentage of the demands rejected among the all requested demands during the simulation:

$$Demand\ rejection\ rate = n\_rej / n\_req \tag{3.7}$$

Where *n_rej* is the number of demand rejected, *n_req* is the total number of demands requested for connection.

Since SP and SWP algorithms are not the QoS solutions, even if any link along the selected path does not have sufficient capacity these two algorithms send the current flow on the shortest or shortest widest path. So they do not have the rejected flows but they have much more lost packets. In contrary, LBWDP has a few packet loss rates and also no rejected flow because it increases the availability of routers through minimizing the maximum link utilization rate.

3.5. Conclusion

In this chapter, we have selected LDM and WDP among existing multipath routing algorithms and we have divided them into modular part of multipath routing scheme presented in chapter 2. As one modular for second step, we have suggested a new traffic splitting method using the prospect method of effective repartition, called PER. With these modular parts, we have proposed three hybrid load balancing algorithms for multipath QoS. The first one combines WDP for step1 with LDM for step2. The second one combines LDM for step1 with new PER method for step2. Finally the third combines WDP for step1 with our new PER method for step2. This last algorithm is called LBWDP for Load Balancing based on Widest Disjoint Paths. When calculating the candidate paths in LBWDP algorithm, we consider constraints such as link capacity (residual bandwidth) for accepting the QoS traffic. The load balancing objective of our three hybrid algorithms is to minimize the maximum link utilization over the network, so we have compared their performances in terms of maximum link utilization ratio.

The statistical performance of the algorithm was studied through simulations. First we have solved TB formulation using MATLAB as an optimal bound for comparison and simulated our three hybrid algorithms and SP, SWP using *ns-2*. By the simulation, when one uses LBWDP algorithm the more is the traffic, the better is load balance in the network in comparing with algorithms such as SP and SWP.

Three hybrids' simulation results showed that they are satisfiable and particularly the third hybrid, LBWDP's load through the network was best balanced so that the network throughput is improved. LBWDP gets better results for load balancing than

other hybrids. We can propose the possibility the multi-model approach which performs different algorithms corresponding to the load.

The experiments done to evaluate the efficiency of the proposed algorithms show that our assumption to use link utilization criterion as the single criterion to implement load balancing is a good choice. Indeed, we show that the more the average link utilization rate is low, the less are the packet loss rate and the demand rejection rate. This result must also be confirmed for the delay.

MPLS offers many advantages to service providers. However, it is incapable of providing differentiated service levels in a single flow. Hence MPLS and DiffServ seem to be a perfect match and if they can be combined in such a way to utilize each technology's strong points and counter the other's weaknesses, it can lead to a symbiotic association that can make the goal of end to end QoS feasible.

In this chapter, we have proposed the routing algorithm for MPLS network. In chapter 4, PER algorithm will be used for supporting the differentiated service levels in the DiffServ-aware MPLS Traffic Engineering (DS-TE) network.

Chapter 4:

Integration of DiffServ and MPLS

4.1. Introduction

MPLS TE (Traffic Engineering based on MPLS) and DiffServ (Differentiated Services) can be deployed concurrently in an IP backbone. DiffServ can provide packets with a preferential treatment using different code-points in their header. MPLS networks can be configured to offer different QoSs to different paths through the network and enables greater control over routing in packet networks. If the two technologies are combined, then standardized DiffServ service offerings can be made and it can provide differentiated services with greater routing control. Such control means that it is more likely the operator will be able to offer services within well-defined QoS parameters. This combination of MPLS and DiffServ is called DS-TE (DiffServ aware MPLS Traffic Engineering) [RFC3564].

In this chapter, we propose new DS-TE model for the intra-domain network, called PEMS (PEriodic Multi-Step algorithm for DS-TE network) [Lee, 2006c], to give the differentiated services for three classes: EF class (Expedited Forwarding), AF (Assured Forwarding) class and BE (Best Effort) class. The EF class requests every router along the path to always service EF packets at any rate as fast as the rate at which EF packets arrive. It is the delay sensitive traffic like VoIP traffic. The AF class requests more flexible and dynamic sharing of network resources by soft bandwidth and loss guarantees suitable for burst traffic like ftp or e-mail. BE class is also classified in order to assure that best effort traffic can have a minimum rate even in the worst case. The EF class should get the best QoS. Differentiated Services may have different traffic characteristics as well as different QoS requirements. Supporting differentiated services

needs to supply network resources, bandwidth or buffer respectively according to the class. Efficient and optimized algorithms for resource provision are needed.

New routing scheme for DS-TE proposed in this chapter aims to obtain different treatments for the aggregated flows according to their traffic classes and load balancing while the LBWDP algorithm (Load Balancing over Widest Disjoints Paths algorithm) objective is to minimize the maximum link utilization to balance the load as presented in the previous chapter. Our goal is to develop a routing method that optimizes differentiation of experienced service of traffic classes in terms of two metrics, delay and bandwidth.

To start, PEMS selects some basic paths to avoid path searching overhead in the big topology. Among them, each class computes its candidate paths based on its appropriate metric and it calculates selective ratio values with different weight of delay and bandwidth for being used in the traffic splitting stage. In the traffic splitting stage, it also adapts PER algorithm (Prediction of Effective Repartition) of LBWDP (Load Balancing over WDP algorithm) with its own candidate paths. So ingress router sends a traffic demand on the better LSP with regard to the metric that characterizes its membership class. It performs the traffic engineering process offering appropriate services for various classes of traffic. Instead of assigning a fraction of each link to each class, our model isolates the various flows on disjoint paths as long as possible inside the network to yield a better quality of service to the various classes.

By simulation, it is proved that each type of traffic can be differentiated and guaranteed by PEMS. PEMS is compared in terms of measured delay and link utilization rate, with LBWDP which does not differentiate traffics' classes.

4.2. DiffServ aware MPLS Traffic Engineering

MPLS technology is a suitable way to provide TE (Traffic Engineering) owing to its path controllability. MPLS can combine with DiffServ to extend MPLS traffic engineering to enable performing routing with class-based constraint. This ability to satisfy a class-based constraint translates into an ability to achieve higher QoS performance in terms of delay, jitter, throughput or packet loss [Pasquale, 2004].

DS-TE mechanisms operate on the basis of different DiffServ classes of traffic to improve network performance and extend the base capabilities of TE to allow route computation and admission control to be performed separately for different classes of service. With DS-TE, it is possible to define explicit routes with different performance guarantees on each route to ensure QoS constraints are met. It can also give network designers the flexibility to provide differential treatment to certain QoS classes that need path-protection and the strict QoS guarantees while optimizing the network resources utilization.

DS-TE mechanisms must decide how to distribute the resources differently to each class. The links can be divided to be used by each traffic class with appropriate rate. Or it is also possible to define different paths for each class. For the former, Bandwidth Constraint model (BC model) is a solution for dividing each link. Normally after that DS-TE system defines some class types to be serviced, an efficient routing algorithm to search the proper LSPs according to classes' requirements must be developed. The system can use one routing algorithm for all classes or different algorithms for each class [Bai, 2003].

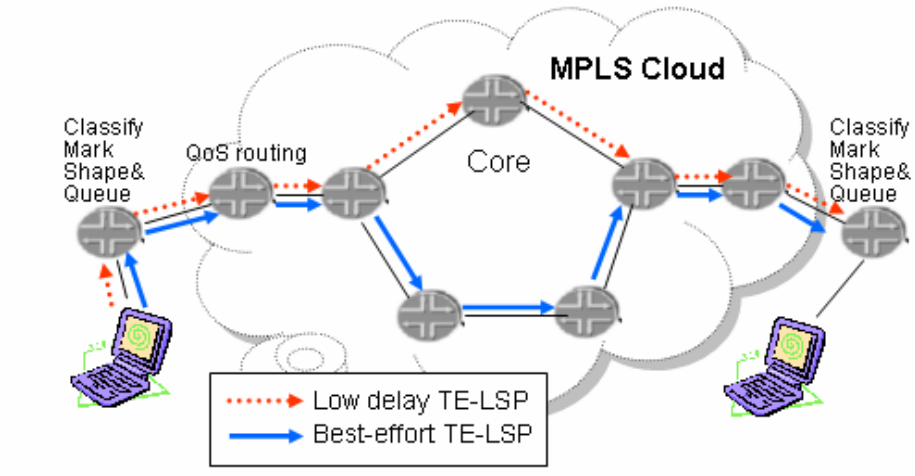


Fig 4. 1 DiffServ-Aware MPLS Traffic Engineering (DS-TE)

Fig 4.1 depicts an example of DS-TE mechanism based on MPLS LSPs. When a traffic demand arrives at an ingress router, it is classified and marked according to its DSCP (DiffServ Code Point) in the packet header. Thus, the ingress calls an appropriate

routing algorithm to find the best path for the current demand. For example, it gives the shortest path (dashed path in the Figure 4.1) for a LSP request which requires low-delay, and the longer and big-capacity path (plain path in Figure 4.1) for best effort traffic. By using different paths according to traffic class, DS-TE can give different quality of service to users and it can use efficiently its network resources.

4.2.1. State of art : Differentiated Service approaches

In recent years, research to give differentiated service according to traffic type has attracted a lot of attentions. So far several researches have proposed many schemes in DiffServ-aware MPLS network.

IETF introduces Service Provider requirements that enable DS-TE in [RFC3564]. Traffic engineering is done at a per-class level instead of the aggregate level. Traffic of a particular class can be mapped to separate LSPs so that the resources of a network are utilized more evenly [RFC3270] [RFC2430].

Some researches tried to prove that integration of MPLS and DiffServ to ensure QoS guarantees for the backbone networks is mutually beneficial to both MPLS and DiffServ. [Srihari, 2001] showed through simulations that if MPLS and DiffServ were to act alone in the backbone networks in the proposed architecture, the benefits gained are much less compared to that of the combination. It compared, MPLS based traffic engineering with single and multiple LSPs in conjunction with DiffServ and concluded that for the case of multiple LSPs, even when UDP traffic is increased to huge levels the TCP throughput remains at its maximum. Two er-LSPs (explicit routed LSPs) are configured, where one is assigned the EF flow and the other is assigned the AF flow. Each type of traffic follows separate LSPs.

In the paper [Pasquale, 2004], the simulation results show that the use of service differentiation offers protection between classes. Excess traffic in one class does not damage the other classes. The results also show that the use of admission control prevents UDP packet loss for Gold and Silver classes like EF and AF classes of PEMS. It proposes that it is possible to approach the Max-Flow Min-Cut theorem [Skiena, 1990] capacity prediction with good QoS for Gold and Silver traffic. Simulation result

proves that the only effect of the dynamic bandwidth management mechanism is to keep the proportional rare differentiation between the Gold and Silver classes.

[Benameur, 2004] simulates two potential selective service protection mechanisms: differentiated service queue managements and implicit admission control. It distinguishes just two traffic classes; premium and best effort. Packets of a new flow are discarded when an estimation of the current bandwidth is below a specified threshold ensuring the quality of flows in progress. Differentiation can be achieved by applying different admission thresholds to premium and best effort flows such that the former are blocked only in extreme congestion. But they treat the TCP-level traffic, so it is not related with MPLS based IP network in which PEMS is interested. Our PEMS model considers the IP packets in the DS-TE network.

DS-TE system needs the respective metrics for differentiating the service, metric for estimation of the network state and metric to select the different candidate paths.

[Saad, 2001] introduces a probe packet approach to measure the delay on established LSPs between ingress/egress edge routers pair. These measurements are then used in a delay estimation algorithm to provide the measure of the expected delay within the path. The simulation results proved that for EF and best effort classes, this model provides better QoS performance. It combined DiffServ technology with traffic engineering over MPLS to offer an adaptive mechanism that is capable of routing high priority IP traffic over multiple parallel paths to meet delay time constraint. They propose a probe packet method to collect delay measurements along several parallel paths. They use them in an end-to-end delay predictor that outputs a quick current estimate of the end-to-end delay.

[Susitaival, 2002] studies a flow allocation that minimizes the mean delay of the network. It defines the paths using the LP (Linear Programming)-optimization and after that allocates traffic using the NLP (Non Linear Programming)-optimization. Then it divides traffic into parts and routes them consecutively using Dijkstra's algorithm [Dijkstra, 1959]. It develops optimization algorithm that differentiates classes in terms of mean delay.

Routing is studied extensively. In MPLS networks, the routing research has concentrated on LSP routing i.e. how to route the LSPs in the network. It can use the same routing algorithm or different algorithm for each class. It is related with the bandwidth constraint model when all traffic classes share common LSPs. To implement that, the system keeps the sub-pool bandwidth database. [Zhang, 2002] and [Kim, 2004a] tried to differentiate the service using its own sub-pool bandwidth in the extent of its configured rate in all links.

Paper [Chpenst, 2001] offers a solution that dynamically determines QoS-constrained routes with a number of demands and routes traffic within the network so that the demands are carried with the requisite QoS while fully utilizing network resources. By using the modified version of Dijkstra's algorithm [Dijkstra, 1959] they provide a solution for dynamical determining QoS-constrained routes while balancing the load on the network.

In [Akar, 2003], primary and secondary LSPs are established between every pair of IP routers located at the edge of an MPLS cloud. Traffic is split between these two paths using an ABR (Available Bit Rate)-like explicit-rate feedback gathered from the network on a per-flow basis.

PEMS uses multiple candidate paths determined using current network state information according to the class dynamically and they are multiple shorter paths in order of its proper metric, calculated every link state update time.

DS-TE system can use the same routing algorithm with different parameters or different routing algorithm for each class to distribute the traffic in separate LSPs according to the class.

Paper [Zhang, 2002] proposes differentiated routings for different traffic classes and shows that these algorithms can improve network resource utilization and QoS satisfaction of all traffic classes better than when using only one routing algorithm for all traffic classes.

[Bai, 2003] explores the proper routing algorithms by simulations to find the best combination of routing algorithms used for different traffic classes. The basic idea is to route different traffic classes according to their respective QoS requirements. As a result,

according to the assumption that the EF class requests minimum delay, it found heuristically by simulations that both the EF class and the BE class can be maximally optimized when BSP (Bandwidth-inversion Shortest Path) and SP (Shortest Path) are used for the BE and the EF class respectively. It treats the traffic in IP layer as our study but it is not concerned with MPLS.

In PEMS, many paths are compared in terms of their operational costs with each class' respective parameters and current link information. Using the same routing algorithm and different weights of parameters in cost function, each class has its own candidate path set selected by appropriate metric and the traffic is distributed using multiple paths for load balancing.

When two classes prefer one LSP in the same time, the system must adapt new preemption policy. When preemption occurred, the lower-priority LSP will be destroyed, and its bandwidth resource is released. The higher-priority LSP obtains the bandwidth resource to establish its path. While the lower-priority LSP release bandwidth, it has to be rerouted by selecting another LSP, but the LSP cannot ensure whether its bandwidth resources will be preempted again or not. If this situation occurred frequently, routers would have superfluous overload and the quality of delivering flow can not be guaranteed. [Ji-Feng, 2003] and [Oliveira, 2004] propose new policies to avoid the pre-emption because of flow priority and load balancing in the MPLS networks.

The results from the simulation in [Ji-Feng, 2003] indicate that adding the policy to Constraint Routing with comparable QoS of each service level is a better to traditional Constraint Routing. Although higher-level flows did not select shortest path to delivery packets, they still reach expectable performance and avoid preemption.

Some researches propose the integrated traffic engineering and management system for DiffServ-over-MPLS services in Next Generation Internet or address the problem of providing a guaranteed end-to-end QoS over multiple DiffServ compliant domains [Chassot, 2006] [Raghava, 2001] [Kim, 2004b] [Caterina, 2004] [Eleni, 2003] [EuQoS].

They are systems for heterogeneous networks i.e. inter-domains while PEMS considers only intra-domain network.

In this thesis, a QoS traffic routing algorithm that considers multiple metrics is presented and selects the best path according to the class and finally routes traffic demand flows through the selected paths using load balancing algorithm PER.

4.2.2. Mapping DSCP to the EXP field [RFC3270]

The DSCP field represents the service level according to service class in the IP packet header but it is not controlled inside an MPLS domain. In MPLS domain, the packets are routed based on the assigned label rather than the original packet header. To support the differentiated service according to each DiffServ class in an MPLS network, MPLS shim header must infer the PHB (Per-Hop Behaviour) information of DSCP.

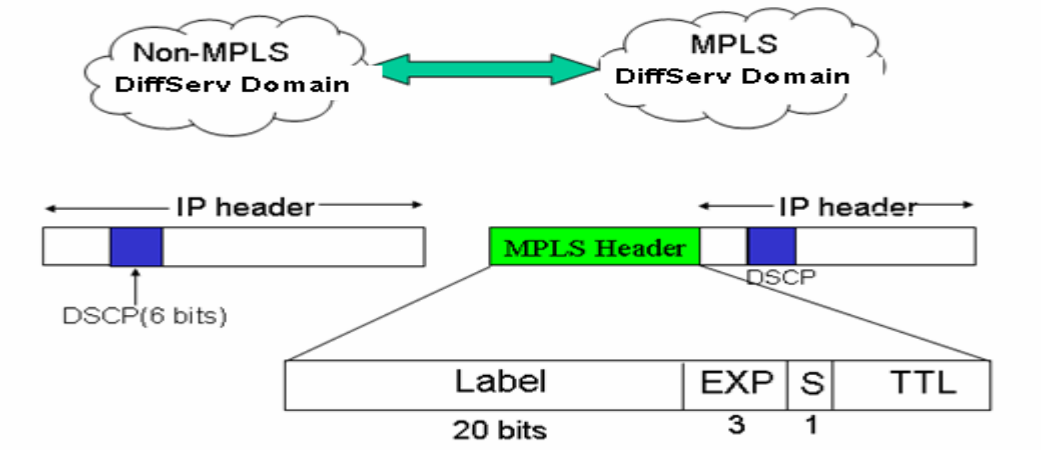


Fig 4. 2 Mapping DSCP to EXP

The IETF proposed to use the three EXPerimental bits (EXP) in the MPLS header to codify the DiffServ information expressed in a 6-bit DSCP in MPLS domain. There are two ways to achieve this, depending on the ways the label header is encoded; E-LSP and L-LSP. PSC represents PHB Scheduling Class.

- E-LSP (Exp-inferred-PSC LSPs): LSR can use the three bit EXP field in MPLS shim header for supporting fewer than eight PHBs. Though the DiffServ

standards allow for up to 64 different DSCPs, MPLS can support only eight different DSCPs when using only the EXP field. If the required number of classes is lower than 8 there is no problem to map the DSCP in IP packet header. The mapping is straightforward: a particular DSCP is equivalent to a particular EXP combination and maps to a particular PHB (scheduling and drop priority).

- L-LSP (Label-only-inferred-PSC LSPs): It is to use the label field itself as the information carrier about different PHBs. L-LSPs can carry packets from a single PHB, or from several PHBs that have the same scheduling regimen but differ in their drop priorities. If there are more classes, the mapping must be more-to-one or the second encoding way must be used. So, using the shim header, the Label field tells to the LSRs where to forward the packet, while the IP packet header tells to the LSRs what PHB to apply to treat the packet. The packets belonging to a common PHB scheduling class must travel on the same LSP [Pasquale, 2004].

In this study, it is assumed that each flow is already classified and it has its own EXP value when it arrives in DS-TE network. It is marked when the LSP demand occurs so our system makes a routing procedure with this class information.

4.2.3. Class-based routing with LSPs

By integration of MPLS and DiffServ, Internet Service Providers can support different quality of services depending on customers' requirements. Two approaches can be considered to implement the service differentiation: sharing of common LSPs by all traffic classes and differentiation of a set of LSPs by traffic class.

4.2.3.1. Sharing of common LSPs by all traffic classes

For simplicity, let us reduce the set of common LSPs to a single one. In this case, the traffic flows to send are subject to DiffServ conditioning, but they are routed

through this single LSP. To send all traffic into the same LSP, all traffic classes must share the LSP.

If one routing algorithm is used for all classes, all traffics can be assigned onto only one LSP until the next link state update. Each type of traffic can be ensured as the amount of its pre-assigned bandwidth of the LSP. Each class can use selected paths at the extent of its configured.

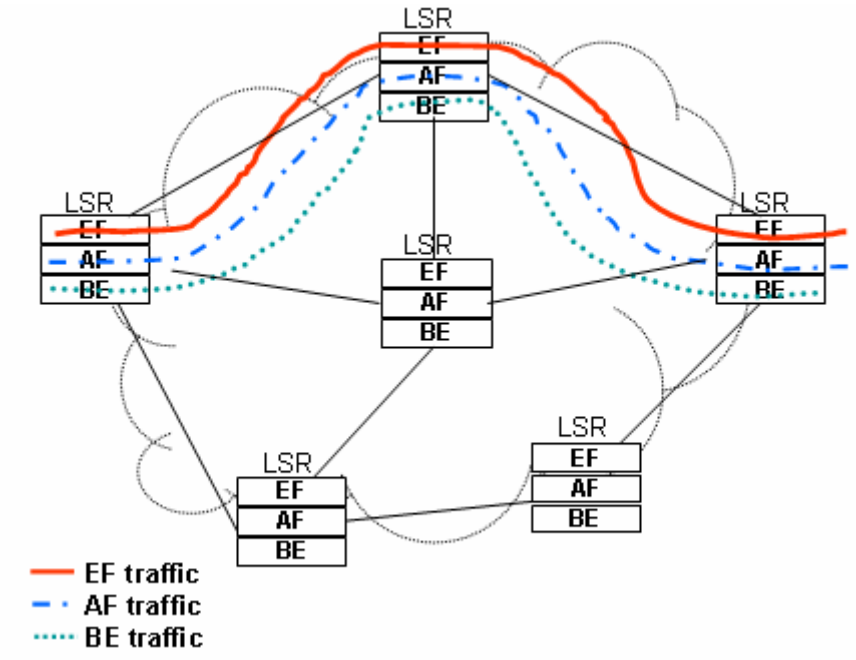


Fig 4. 3 Sharing of common LSPs by all traffic classes

[Srihari, 2001] showed through simulation that MPLS does not introduce any significant advantages to DiffServ networks if a single LSP is used to route all types of traffic.

4.2.3.2. LSP differentiation based on traffic class

Another method to give different quality of service by DS-TE is to provide separate LSP(s) for each traffic class. DS-TE with multiple LSPs uses different paths depending on the level of QoS expected by the traffic to use explicit routing. For selecting the candidate paths to split the traffic, DS-TE model can use different routing algorithm [Bai, 2003], or the same routing algorithm for each class.

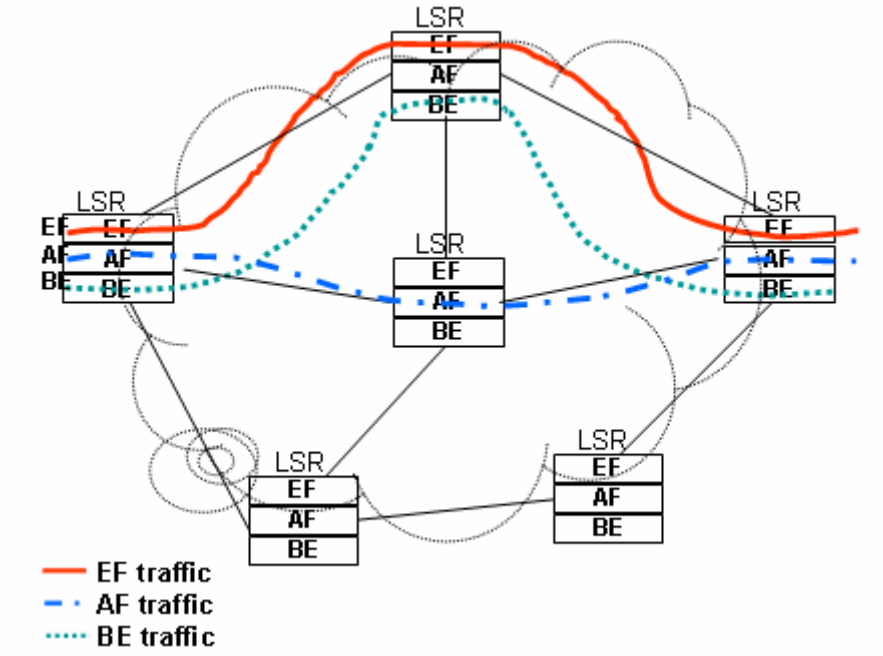


Fig 4. 4 Multiple LSPs for different classes with divided links

To support DS-TE for each traffic class, all traffic classes can share the LSP. Each class of service is associated to an LSP with its configured rate. For this sharing of LSPs, the system must maintain each class' sub-pool of bandwidth as much as pre-configured by administrator. The physical network is divided into multiple virtual networks, one per class. In a per-class type virtual network, multiple QoS-guaranteed TE-LSPs are established among edge routers to configure connectivity of full mesh topology.

When a QoS-guaranteed connection setup (or per-class-type packet flow registration) request is arrived, the connection management module must check the available network resource, find appropriate route for the requested QoS and traffic parameters using own routing algorithm, and finally make decision on the admission of the requested connection establishment.

When the traffic of a class requires LSP, it can compute an appropriate LSP based on its own sub-pool resource. Using Bandwidth constraint model, the link bandwidth is shared between all traffics. Instead of using a fixed repartition of MAM [RFC4128], the amount allocated to one traffic class can be adjusted according to the current load of the

others to utilize the unused resources more efficiently like RDM [RFC4128] or MAR [RFC4126] bandwidth constraint model which will be explained in 4.2.3.4.

As another method for using multiple LSPs, DS-TE system can manage one resources' pool without dividing the links to provide a separate LSP for each traffic class directed to the same egress LSR. It sends the traffic on the best LSP selected for supporting the traffic differentiation of each class but each class does not have its own fraction on the links.

To support this method, DS-TE system can use the different routing algorithms for each class or the same routing algorithm with different parameters like PEMS to select the best LSP to carry the current demand.

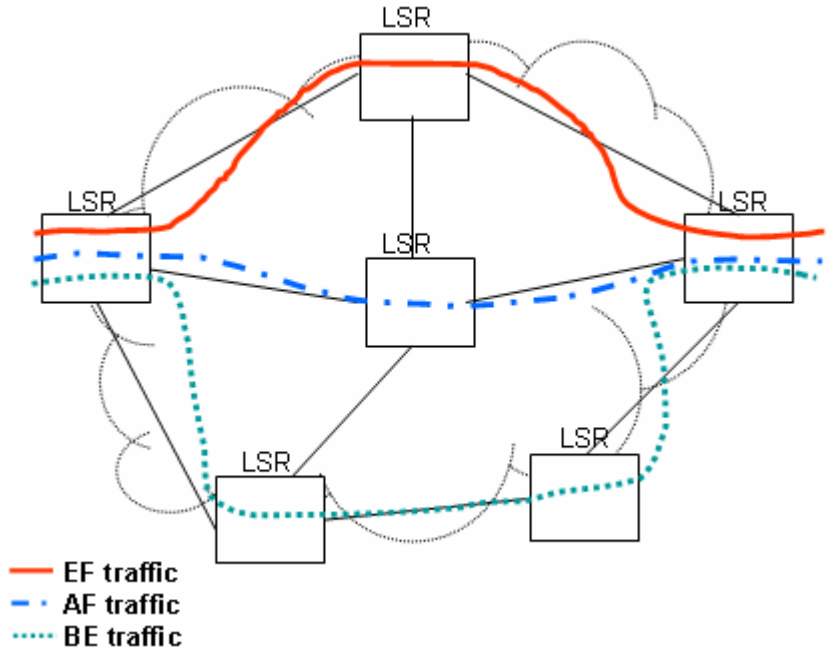


Fig 4. 5 Multiple LSPs for different classes without link fraction

Notably, if different path lengths are present, the best shortest path is reserved to the higher priority traffic. Three LSPs can be constructed with one ER-LSP carrying only EF traffic, one ER-LSP carrying only AF traffic and one ER-LSP carrying only BE traffic. Or it is possible to map one LSP with two classes when that path is best for both classes. PEMS permit this sharing of LSPs. But better paths can be used for low priority class even when higher class needs to reserve those paths. It can be repeated preemption and rerouting in the worst case. DS-TE needs some mechanisms to avoid that

phenomenon, for example the higher priority classes can select first the better paths to reserve than the lower priority classes. In PEMS, multiple paths can be used for each class instead of complicated preemption mechanism.

In this thesis, it is assumed that a traffic flow should be routed on a single path, without splitting. Many flow requests may involve traffic that is inherently unsplittable, and therefore it is important to route them on a single path. Thus, for each flow request, the algorithm must find a path with desired amount of bandwidth between the ingress and egress nodes, or determine that the flow is unroutable.

PEMS model uses multipath routing and does not divide the paths to send all types of traffic. It searches the best path with different weight of metrics for each demand according to the class using current network state information. The paths do not have the fixed repartition rate of each class.

4.2.3.3. Bandwidth Constraint models

For class-based routing each class can have some fraction on the links or share the LSP without fraction. For the first case, all links are divided according to administrative policy for different classes and the same or different routing algorithm can be used for each class as explained in 4.2.3.1. IETF proposes three bandwidth constraint models that constraints the LSP bandwidth; *Maximum Allocation Model (MAM), Russian doll model (RDM) and Maximum allocation with reservation model (MAR)* [RFC4126] [RFC4128].

DS-TE enables the distribution of all classes' load over all available classes' capacity making optimal use of available capacity. To achieve per-class traffic engineering rather than on an aggregate basis across all classes, DS-TE enforces different Bandwidth Constraints (BCs) on different classes. It also provides a tool for constraining the EF class utilization per link to a specified maximum thus providing a mechanism to help bound the delay and jitter.

It is understood that setting an upper bound on the EF class' effective utilization per LSP allows a way to restrict the effects of delay and jitter due to accumulated burst. DS-TE can be used to assure that this upper bound is not exceeded.

It is commonly used a ClassType (CT) to call the different classes in Internet Drafts of bandwidth constraint models [RFC]. CT is necessary to perform the calculation of the available resources. One of the most important aspects of the available bandwidth calculation is the allocation of bandwidth among the different CTs. The percentage of the link's bandwidth that a CT (or a group of CTs) may take up is called a bandwidth constraint (BC). The bandwidth constraints model provides the rules to support the allocation of bandwidth to individual CT.

- *Maximum Allocation Model (MAM)*: This most intuitive bandwidth constraint model maps one BC to one CT. The maximum allowable bandwidth usage of each class, together with the aggregate usage across all classes, is explicitly specified. The problem with MAM is that because it is not possible to share unused bandwidth between CTs, bandwidth may be wasted instead of being used for carrying other CTs.

- *Russian Doll Model (RDM)*: It improves bandwidth efficiency over the MAM model by allowing CTs to share bandwidth. Specification of maximum allowable usage is done cumulatively by grouping successive priority classes recursively. The advantage of RDM relative to MAM is that it provides efficient bandwidth usage through sharing. The disadvantage is that there is no isolation between the different CTs. Preemption must be used to ensure that each CT is guaranteed its share of bandwidth no matter the level of contention by other CTs.

  Suppose that the three classes of traffic are denoted by class 1 (EF class, highest priority), class 2 (AF class), and class 3 (BE class, lowest priority). When preemption is enabled, these are the preemption priorities. To define a generic class of BC models, let

```
Nmax = path capacity; i.e., the maximum number of simultaneously
       established LSPs for all classes together

Nc = the number of simultaneously established class c LSPs,
     for c = 1, 2, and 3, respectively.
```

  For MAM, let

```
Bc = maximum number of simultaneously established class c LSPs.
```

Then, *Bc* is the Bandwidth Constraint for class *c*, and we have

```
Nc <= Bc <= Nmax, for c = 1, 2, and 3
N1 + N2 + N3 <= Nmax
B1 + B2 + B3 >= Nmax
```

For RDM, the BCs are specified as:

```
B1 = maximum number of simultaneously established class 1 LSPs
B2 = maximum number of simultaneously established LSPs for
classes
      1 and 2 together
B3 = maximum number of simultaneously established LSPs for
classes
      1, 2, and 3 together
```

Then, we have the following relationships:

```
N1 <= B1
N1 + N2 <= B2
N1 + N2 + N3 <= B3
B1 < B2 < B3 = Nmax
```

- *Maximum allocation with reservation model (MAR)*: Bandwidth allocated to individual CTs is protected as needed, but otherwise it is shared. Under non-congested network conditions, all CTs fully share all available bandwidth. When congestion occurs for a particular CTc, bandwidth reservation prohibits traffic from other CTs from seizing the allocated capacity for CTc. It is similar to MAM in that a maximum bandwidth allocation is given to each CT. However, through the use of bandwidth reservation and protection mechanisms, CTs are allowed to exceed their bandwidth allocations under conditions of no congestion but revert to their allocated bandwidths when overload and congestion occurs [RFC 4126, Jun. 2005]. On a given link *k*, a small amount of bandwidth *RBW_THRESk* (the reservation bandwidth threshold for link *k*) is reserved and governs the admission control on link *k*. Also associated with each CTc on link k are the allocated bandwidth constraints *BCck* to govern bandwidth allocation and protection. The reservation bandwidth on a link (*RBW_THRESk*) can be accessed when a given CTc has bandwidth-in-use (*RESERVED_BWck)* below its allocated bandwidth constraint (*BCck*). However, if *RESERVED_BWck* exceeds its allocated bandwidth constraint

(*BCck*), then the reservation bandwidth (*RBW_THRESk*) cannot be accessed. In this way, bandwidth can be fully shared among CTs if available, but is otherwise protected by bandwidth reservation methods. Bandwidth can be accessed for a bandwidth request = *DBW* for CTc on a given link *k* based on the following rules:
For LSP on a high priority or normal priority CTc:
If $RESERVED\_BWck \leq BCck$: admit if $DBW \leq UNRESERVED\_BWk$
If $RESERVED\_BWck > BCck$: admit if $DBW \leq UNRESERVED\_BWk - RBW\_THRESk$;
or, equivalently:
If $DBW \leq UNRESERVED\_BWck$, admit the LSP.
For LSP on a best-effort priority CTc: allocated bandwidth $BCck = 0$;
Diffserv queuing admits BE packets only if there is available link bandwidth. The normal semantics of setup and holding priority are applied in the MAR Bandwidth Constraints Model, and cross-CT preemption is permitted when preemption is enabled.

The internet draft [Faucheur, 2002] concludes through the comparison of two BC models, MAM and RDM, that the Russian Dolls Model matches very well the canonical DS-TE objectives in the canonical DS-TE deployment scenario (as well as many other practical deployment scenarios) and it recommends selecting the Russian Dolls Model as the default model for DS-TE. Based on this recommendation, DS-TE supporting routers can decide one between RDM and MAR according to administrative objective.

But PEMS manages one resource pool for traffic engineering and sends the demand onto the best LSP with regard to its class without configuring the links for sharing between classes. It makes the efficient routing according to the class without the bandwidth constraint model.

4.3. Proposition of Periodic multi-step algorithm for DS-TE network (PEMS) [Lee, 2006c]

PEMS has three steps to select one best path to assign current demand. Of three, pre-processing phase is done with only network topology in offline mode; the candidate path selection phase and the splitting phase to reserve it for current demand are done in online mode. Specifically, candidate paths selection phase is taken every link state update time and the splitting phase is done at each demand arrival.

4.3.1. General mechanism

Efficient routing of user request is essential in a DiffServ/MPLS network to satisfy QoS and manage network resources. As proved in [Srihari, 2001], it is better to use differentiated LSPs for different traffic classes. What is important is to select the better LSPs specialized for each class while ensuring the efficient resource utilization. Our proposition focuses on isolation of the various flows on the better paths as long as possible inside the network yields in a better quality of service to the various classes. LSPs for different classes can have different constraints. It starts to find the maximally link disjoint paths and assigns the low-delayed path to EF class to especially ensure its quality.

The three phases of PEMS are as follows;

- Pre-processing stage – Offline mode
- Online candidate paths computing stage – Online mode
- Online demand splitting stage for LSP requests – Online mode

In the pre-processing phase, it extracts good paths of all possible paths which can include every link at least once within them for each source-destination pairs using only topology information, in the offline mode. These paths are kept until the topology is changed. To select these basic paths, we propose new algorithm which finds some maximally link disjointed paths in 4.3.2.2. This using of disjointedness is for reducing the congestion probability.

In the online mode, when link state information are updated, new candidate paths for each class are calculated based on updated information such as measured delay and residual bandwidth. At this point, we use metric ordering by delay and residual bandwidth. This phase selects multiple low-delayed paths in the ordered paths set as candidate paths of delay-sensitive traffic and selects multiple paths having more residual capacity for the traffic to which the bandwidth is important for multipath routing to each traffic class.

When a traffic demand arrives, it uses PER algorithm as proposed in chapter 3 to select one LSP to carry current flow. We adapt the parameters with different weights of delay and bandwidth depending on the class of requested traffic when calculate the selection probability.

Many QoS metrics such as hop count, available bandwidth and delay constraints are considered before the path selection to assign. In PEMS, hop-count and disjointedness are used in the pre-processing phase and available bandwidth, measured delay are used in the cost function to establish splitting ratios.

PEMS aims to minimize the maximum link utilization like LBWDP algorithm basically, and additionally to give different service quality to each class, especially to guarantee the low delay to EF class.

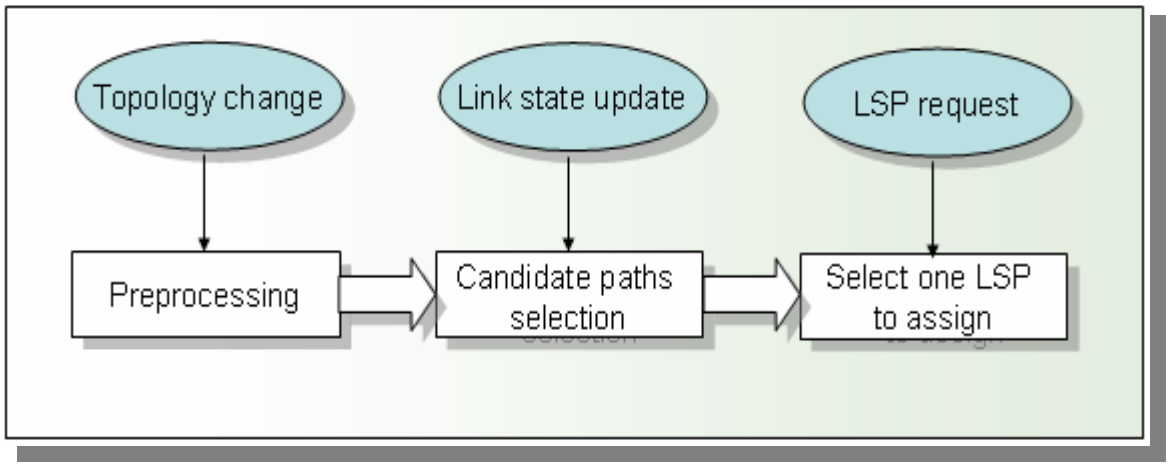


Fig 4. 6 Three stages of PEMS

4.3.2. Routing Procedure

4.3.2.1. Traffic classes

The differentiated architecture is based on a simple model where traffic entering a network is classified and possibly conditioned at the boundaries of the network, and assigned to different PHB that are a collection of packets with common characteristics. Each PHB is identified by a single DSCP. Within the MPLS network, packets are forwarded according to the EXP value associated with the DSCP.

In this study, PEMS distributes three classes EF, AF and BE class for real time traffic, web traffic and best effort traffic, respectively. It classifies best effort traffic into BE class in order to assure that best effort traffic can have a minimum rate even in the worst case.

The goal is to find routes in the network to send as much of each application as possible from its source node to destination node. Service differentiation in DS-TE network effectively preserves the quality of premium traffic, EF traffic.

4.3.2.2. Preprocessing stage

The main objective of using multipath routing is to use several good paths to reach a destination from a source (not just one best path), without imposing excessive control overhead in maintaining such paths. As explained in chapter 1, the link disjointedness is a qualified metric to get some good paths because this characteristic can reduce congestion probability in the network. While multipath routing algorithms can tolerate network failures well, their failure resilience only holds if the paths are selected judiciously. In particular, if paths have some redundant links and any link makes a failure, then other paths sharing this failed link are influenced by this failure. For this reason, to find some mutually disjointed paths is the best but their number is limited by the size of the network.

For the hybrid approach of chapter 3, WDP algorithm is used to implement the candidate path selection part, which selects some good disjoint paths using their path widths based on residual bandwidth. It has a restricted number of paths but it does not have enough paths to support our model which needs more paths as candidate paths for

three classes, so our model needs bigger topology or new algorithm with the smoothen conditions to obtain more candidate paths. The most expensive part of our algorithm is this preprocessing phase, which is run very infrequently and offline, only when the network changes. This is a prerequisite part for the operation of PEMS.

A new method to find multiple paths is proposed to select some maximally link disjointed paths. This new algorithm determines a set of some good paths called "*basic paths*", in order to reduce the computation overheads to check all paths in the online mode. The set of basic paths between an ingress node '*i*' and an egress node '*j*', BP_{ij}, must be as short as possible and must have the minimum number of shared links.

In the literature, there are several types of disjoint paths. Two node disjoint paths have no common nodes except the source and destination nodes. Two edge disjoint paths have no common edges. Two maximally disjoint paths mean that among some set of choices the maximally disjoint paths share the fewest nodes or edges in common. We find maximally edge disjoint paths with least number of links jointed for each pair of source-destination nodes.

New method starts calculating all possible paths P_{ij} using K shortest paths algorithm [Eppstein, 1998] which calculates a set of paths between two given nodes, which are ranked in ascending order with respect to path length. The shortest paths of P_{ij} are the only elements of the basic path set BP_{ij} at the start time. Our algorithm performs all paths checking task, which determines whether a path can be accepted as basic paths, by checking its path length and its links shared with the paths in BP_{ij}. Naturally it finds mutually disjoint paths firstly because it tries to find the paths which shared links with the shortest path is zero. And if there are no paths which do not have shared links with the shortest path, the permitted number of shared links is increased by 1 to find other basic paths. It is repeated until all links are included at least once for generating BP_{ij}. At most one alternate path is generated at each execution of the algorithm. The previously determined paths' resources are transformed for the next iteration. To determine k paths, the algorithm is executed iteratively at least $k-1$ times because the shortest path does not need any iteration. Procedure is ended when all links in the network are used at least once for generating BP_{ij} in order to utilize all links for load balancing.

The complexity of the algorithm is $O(M^3)$ with M the number of the network links.

- **Inputs** - The input graph $G = (V, E)$, V is nodes' set and E is links' set
  - Source node i and destination node j
- **Output** - Basic paths set, BP_{ij}, from i to j, such that they are maximally link disjointed

Path Selection Algorithm *Maximally disjoint path retrieve* (*G, i, j*)

Step 1. /* Initialization */
$P_{ij} \leftarrow$ all paths *(G, i, j)*
$s \leftarrow 0$ /* s is the number of shared links */
$ls \leftarrow$ links_of (P_{ij})
$BP_{ij} \leftarrow$ shortest path of P_{ij}
Step 2. do
Step 2.1 $l_{cp} \leftarrow$ links_of (BP_{ij})
Step 2.2 $P'_{ij} \leftarrow P_{ij} - BP_{ij}$
Step 2.3 $P_d \leftarrow$ next_path (P'_{ij})
Step 2.4 If ($P_d \neq \emptyset$) then
add $\leftarrow$ *true;*
$l_{Pd} \leftarrow$ links_of (P_d)
For each *p(k)* in BP_{ij} do
lp(k) $\leftarrow$ *links_of (p(k))*
If ($card(l_{Pd} \cap lp(k) \geq s)$ then
add $\leftarrow$ *false* ; endif
End foreach
If *(add = true)* then
$BP_{ij} \leftarrow BP_{ij} \cup \{P_d\}$
Else goto Step 2.3
Else s $\leftarrow$ s + 1
while (*ls <> lcp)*
Step 3. return BP_{ij}

This algorithm has heavy complexity but it is not important because it is executed offline and only when topology is changed.

For a better comprehension of this path searching algorithm, an example comes to explain step by step with a simple topology. Suppose that source router is *node1* and destination router is *node7*.

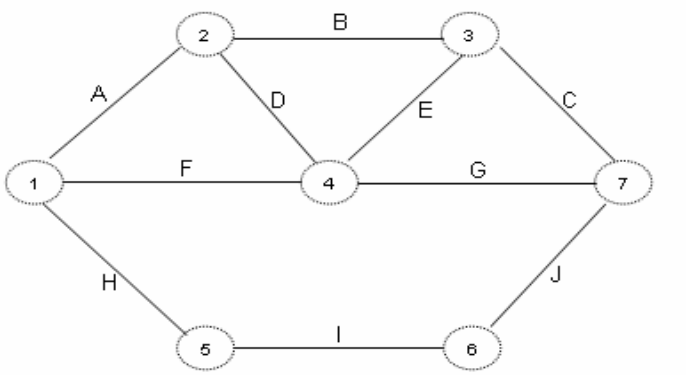


Fig 4.7 Example Topology

P_{ij} is the set of all possible paths without link redundancy in order of link count.

- P_{ij} = {FG, ABC, HIJ, ADG, FEC, ADEC, FDBC, ABEG} → Step1 .

The basic path set *BP* starts from the shortest path {FG} and *s* is 0.

- BP_{ij} = {*FG*} *ls* = { *A, B, C, D, E, F, G, H, I, J*} *s = 0* → Step1
- l_{cp} = {*F,G*} → Step2.1

It finds the next path that does not include two edges in the set l_{cp}, *F* and *G*. It is *ABC*.

- BP_{ij} = {*FG, ABC*} → Step2.4.
- l_{cp} = {*A, B, C, F, G*} *s = 0* → Step2.1.

It continues to find the next path that does not include the edges in the set l_{cp}. It is *HIJ*.

- BP_{ij} = {*FG, ABC, HIJ*} → Step2.4.
- l_{cp} = {*A, B, C, F, G, H, I, J*} *s = 0* → Step2.1.

Repeatedly it tries to find the next path but there is no satisfied path. So *s* is increased which permit to one more edge redundant.

- *s = 1* BP_{ij} = {*FG, ABC, HIJ*} → Step2.4.
- l_{cp} = {*A, B, C, F, G, H, I, J*} → Step2.1.

It finds ADG. It appends two edges *D, G* to the set l_{cp}.

- *s = 1* BP_{ij} = {*FG, ABC, HIJ, ADG*} → Step2.4.
- l_{cp} = {*A, B, C, D, F, G, H, I, J*} → Step2.1.

It cannot find the next path with $s = 1$ so s is increased again, $s = 2$ (Step2.4). It means that it finds two-edge redundant path with set l_{cp}. s is increased until l_{cp} is equal to all links ls. The result path is FEC.

- $s = 2$ $BP_{ij} = \{FG, ABC, HIJ, ADG, FEC\}$ → Step2.4.
- $l_{cp} = \{A, B, C, D, E, F, G, H, I, J\}$ → Step2.1.

At this time, set l includes all links in the network, $ls = l_{cp}$(Step 2), the algorithm is terminated. The selected five paths are marked in Fig 4.8.

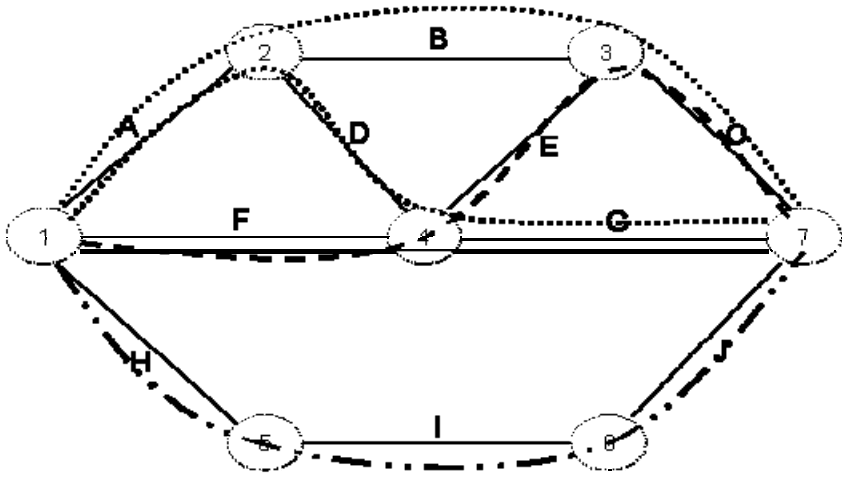


Fig 4.8 Selected LSPs from the algorithm

Some paths to be used in next online candidate path selection stage are obtained by this preprocessing stage. Preprocessing stage is needed to limit the number of paths, because the online candidate path selection calculation from the set of all possible paths for each source destination pair is too expensive and too exhaustive. Thus this algorithm finds multiple paths, maximally disjoint paths with minimum total length between a pair of nodes. But this algorithm is so expensive that it is suitable for networks whose topology changes not frequently. One can ameliorate this algorithm by fixing the parameter for the number of permitted shared links according to objectives.

4.3.2.3. Candidate paths computing stage

This stage starts with the preprocessed basic paths for each source-destination nodes, given by the previous stage. This stage is executed every link state update time and it calculates the appropriate candidate paths set for each class, CP_{EF}, CP_{AF} and CP_{BE} from the set of pre-selected maximally disjointed path set, BP_{ij}.

In PEMS, the residual bandwidth and measured delay are used as two metrics for selecting their candidate paths and for calculating their splitting ratio. Some information are kept in Traffic Engineering database by population through information obtained from the routing flooding mechanisms about link capacities, unreserved capacity, measured delay and so on. On changing state information, all paths of BP_{ij} from the preprocessing phase are sorted by delay and residual bandwidth, for EF and AF class respectively. And then, for delay-sensitive EF traffic PEMS selects some lower-delayed paths, CP_{EF}, and for AF traffic it selects paths owning more residual bandwidth as its capacity is important in the sorted paths, CP_{AF}. The number of candidate paths is decided depending on the topology size or the administrative policy. Since the topology used for our simulation is not too big, we put three as the number of candidate paths of EF and AF classes heuristically. It is possible that some paths can be included in the two sets at the same time. And then BE class considers the remaining paths, CP_{BE}, in the BP_{ij}. The goal of this stage is to choose feasible paths for differentiating the quality of service according to the class and to achieve efficient utilization of network resources by multiple paths selection.

When candidate paths for each class are decided, each candidate path's splitting ratios are also calculated in this stage. These ratios are used to perform the PER algorithm in the traffic splitting stage. Candidate paths are compared in terms of their operational costs. The cost considers two important metrics for the path selection. The factors pertaining to the different metrics are weighed by their corresponding important factor which can be varied depending on the class. To calculate the splitting ratio of each path, our model uses the cost function (4.2) similarly as LBWDP algorithm in Chapter 3. But it has two differences in that PEMS uses measured delay $de(i)$ instead of

hop-count and that we adapt different p_0, p_1 values according to the class unlike LBWDP which puts the same weight to two parameters.

de(i) is the measured delay of *LSPi* and *D* is a constant to make the sum of the probabilities that are inversely proportionate to delay of an LSP_i, *de(i)*. *D* is calculated by the equation (4.1).

$$D = \frac{1}{\sum_{i=1}^{k} \frac{1}{de(i)}} \tag{4.1}$$

b(i) and *B* are calculated same as LBWDP algorithm. The splitting ratio of LSP_i, r_i, is the preference value of LSP and is calculated with the equation (4.2).

$$r_i = p_0 \frac{D}{de(i)} \quad + \quad p_1 \frac{b(i)}{B} \quad with \quad p_0 + p_1 = 1 \tag{4.2}$$

Bandwidth is associated with delay in this model for differentiating the traffic at the flow level. Bandwidth has a bigger weight (parameter p_1) for AF class while delay has a bigger weight (parameter p_0) for EF class. Adaptation of selective parameters is to give different weight according to the metric important of each class. PEMS puts the weight parameters, p_0 and p_1, of each class as follows.

Class	EF class	AF class	BE class
p_0	0.7	0.3	0.5
p_1	0.3	0.7	0.5

For EF class, p_0 is bigger than p_1 in order to give preference to LSP in BP_{ij} that owns the best delay than residual bandwidth because this class is for delay-sensitive traffic. For AF class, the criterion is inversed and so parameter p_1 is greater to express the preference of LSP with important residual bandwidth.

This stage can be ameliorated by adapting dynamically the parameters of the splitting ratio equation depending on the network state.

4.3.2.4. Online splitting for LSP requests

It is necessary to define a mapping between the DiffServ classes and the LSPs in the DiffServ/MPLS network to achieve efficient resource utilization. This will provide better resource utilization by performing traffic engineering at DiffServ level.

This stage provides a framework to choose the most efficient path in the proper candidate path set for the traffic flow. For each traffic class, the candidate paths are known beforehand every link state update time. The model tries to distribute the traffic of different classes respectively to their own candidate paths in an optimal way according to PER algorithm.

Each network node floods information about its state to the whole network at periodic time intervals. This results in partial information about the network state at each node. The information is partial because the network nodes do not have current information about the complete network state. Instead, they have information for the time instant when the last update was generated. When a path selection request arrives, an estimation and forecast algorithm like PER can be used to obtain more accurate information about the current network state.

When a traffic demand is arrived in the source router, routing algorithm must select one LSP to assign that. For that, our model uses PER algorithm proposed in chapter 3 but we calculate the selection probability S_i using its own candidate paths set and their splitting ratio r_i calculated in the candidate path calculation stage using their own parameters.

Each LSP request specifies the amount of bandwidth requested and the class of service. The traffic demand does not have delay demand but only traffic class. PEMS supports the best service in terms of delay constraint to EF class. Based on the information in the request, PER algorithm tries to locate the LSP that best meets the requests using a selection probability. PER algorithm balances traffic load among the candidate paths by assigning arriving flows for selection probability after assigning to become closer with pre-calculated one.

PEMS finds low-cost feasible paths that have sufficient resources to satisfy the QoS requirements of a request while restricting the number of hops and the delay on the path if needed according to the class.

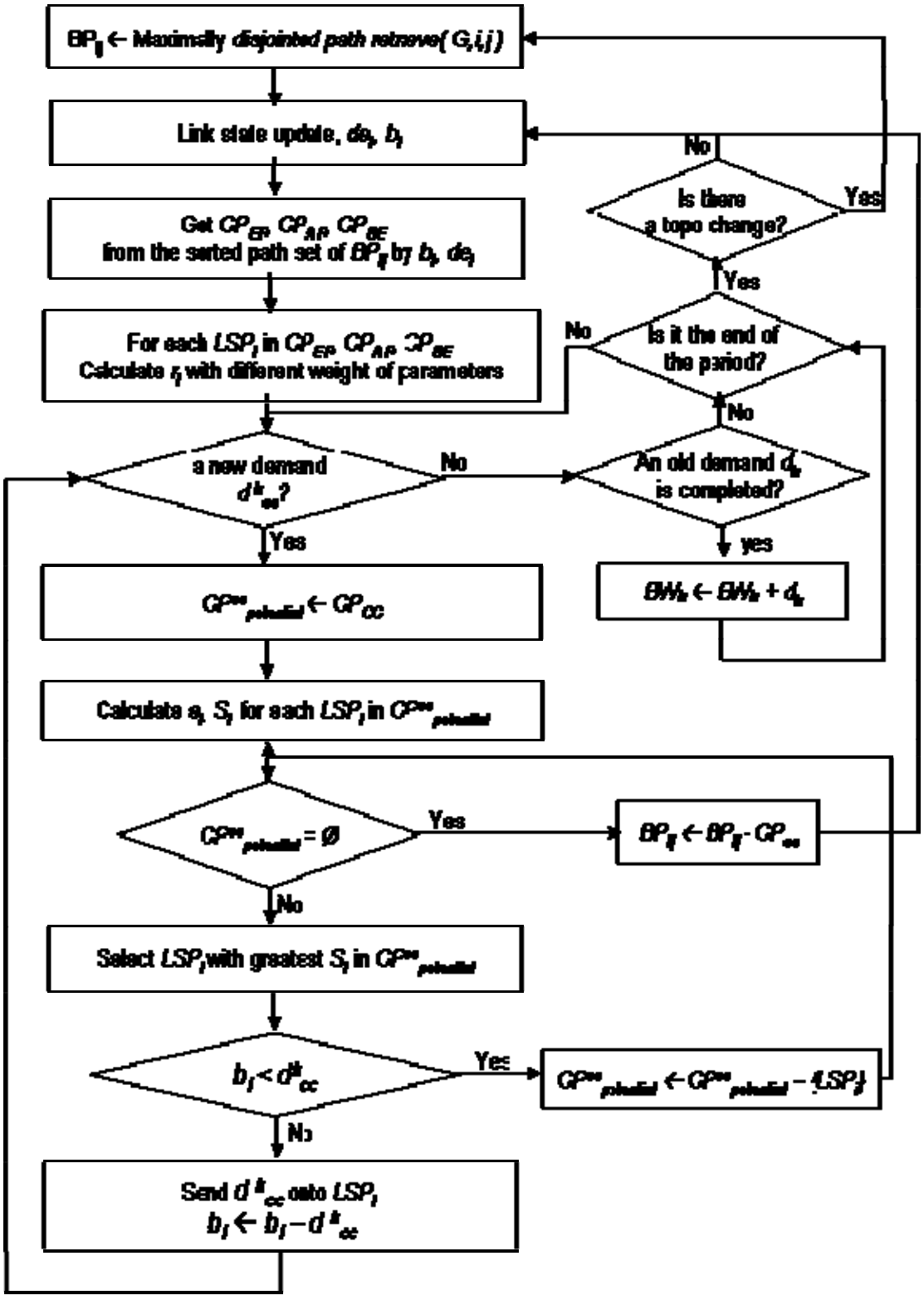


Fig 4.9 PEMS flowchart

Fig.4.9 depicts the flowchart of PEMS. The notations used in Fig. 4.9 are as follows;

- de_i : delay of LSP_i
- b_i : residual bandwidth of LSP_i
- CP_{EF}, CP_{AF}, CP_{BE} : candidate path set for EF class, AF class and BE class respectively
- d^k_{cc} : k-th demand with class cc
- CP_{cc}: current class (one in CP_{EF}, CP_{AF} or CP_{BE})
- $CP^{cc}_{potential}$: subset of CP_{cc} corresponding to LSP_i that can process the requested demand d^k_{cc}

4.4. Performance study

PEMS' objective is to obtain different treatments to meet the customer's requirements for the aggregated flows according to their traffic class ensuring minimization of the maximum link utilization for load balancing as LBWDP's objective in the previous chapter.

The goal of the simulation in this subchapter is to show that our proposed scheme can differentiate the quality of service according to the class and that there is a benefit in combining MPLS and DiffServ in the backbone networks. This goal is achieved through performance comparison between PEMS and LBWDP acting in a network under the same network traffic conditions, to know whether PEMS can arrive at the service differentiation without the big damage of load balancing.

There are different commonly used performance metrics like throughput, utilization rate, reliability and availability. For our simulation experiments, end-to-end delay and link utilization are chosen to be the performance metric of interest. The link utilization which is good to be a performance metric as explained in chapter 3 is selected to show whether the network load is balanced well. End-to-end delay is added as a new performance metric in this simulation to estimate whether delay-sensitive traffic, EF traffic, can be guaranteed its appropriate service.

For simplicity, we assume that there is a traffic engineering server that knows the current network topology and available link capacities. Instead of dedicating a single

machine to perform route computations, this job could also be shared among all ingress nodes without changes to the framework.

4.4.1. Simulation topology and traffic scenario

Since multiple classes use multipath routing, our system needs sufficient LSPs to support that. The simulation results presented below are based on experiments conducted using the topology as shown in Fig.4.10 to have sufficient LSPs to support DS-TE multipath mechanism.

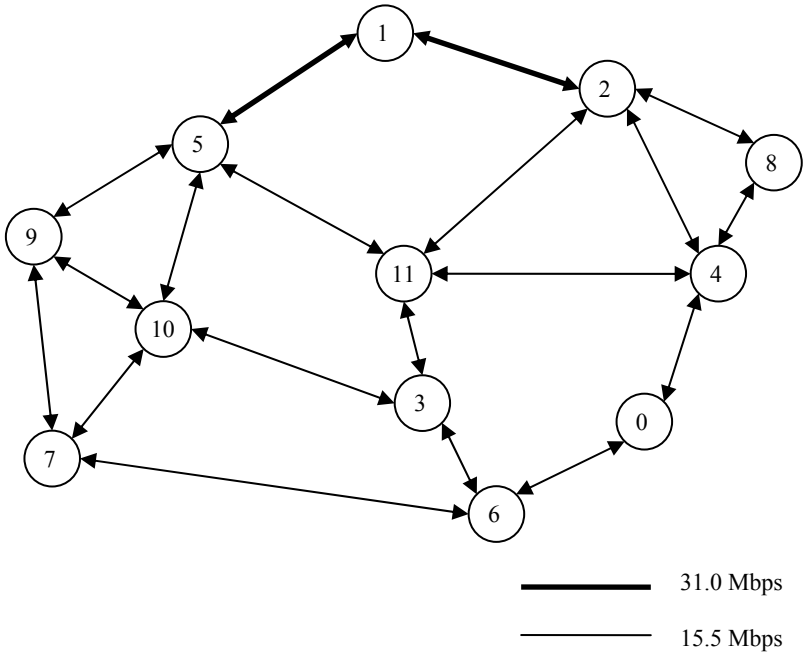


Fig 4.10 Simulation topology for PEMS

Suppose that possible source-destination pairs are LSR0-LSR9, LSR4-LSR7 and LSR8-LSR5. With this topology, our pre-processing phase gives eight paths for the pair LSR0-LSR9, six paths for the pair LSR4-LSR7 and seven paths for the pair LSR8-LSR5 by maximally link disjoint path searching algorithm explained in 4.3.2.2.

LSR0-LSR9	LSR4-LSR7	LSR8-LSR5
0-6-7-9	4-0-6-7	8-2-1-5
0-4-11-5-9	4-11-3-10-7	8-4-11-5
0-6-3-10-9	4-2-1-5-9-7	8-2-11-3-10-5
0-4-2-1-5-9	4-11-3-6-7	8-4-0-6-7-9-5
0-6-7-10-9	4-2-11-5-10-7	8-2-4-11-5
0-4-8-2-11-5-9	4-8-2-11-3-6-7	8-4-0-6-3-10-5
0-4-11-3-10-9		8-4-0-6-7-10-9-5
0-4-11-5-10-9		

Every link state update time, it calculates each class' candidate paths. The paths are chosen based on each class' metric important for the candidate path selection such as link available bandwidth, measured delay. In this study we have heuristically limited to three the number of candidate paths in EF and AF classes. PEMS finds better feasible paths that have sufficient resources to satisfy the QoS requirements of a request while restricting the number of hops and the delay on the path if needed according to the class.

For EF class it selects top three in the paths sorted by ascending order of measured delay, for AF class it selects top three in the paths sorted by descending order of measured residual bandwidth, and it takes the remaining paths for BE class.

Since measured delays of all paths in basic paths BP_{ij} are 0 at the start time, candidate paths for EF class are three shorter paths notably. AF class takes three in the order of total capacity in the beginning. From the second flow request, it can use the measured delay and residual bandwidth to select EF and AF class' candidate paths respectively. Delay is measured dynamically for each demand. But the residual bandwidth is kept and used as the same for one period of link state update and that's why temporary resource database like in PER is needed. The candidate paths for BE class are the remains except paths selected for EF and AF classes.

And then LSP request must select one best path to be assigned among requested class' candidate paths. Our method is to use the traffic engineering features of MPLS in

combination with DiffServ conditioning to provide a separate LSP for each traffic class. A request for an LSP setup is defined by a quadruple *(src, dst, d_k, cc)*, which *src* is the ingress router, *dst* is the egress router, d_k is the bandwidth requested for the LSP and *cc* is a traffic class.

The algorithm can either accept the request, in which case it must find a path in the network from *src* to *dst* along which each link has residual capacity at least d_k, or the algorithm may reject it. We assume that all LSP setup requests arrive online, one at a time, and the algorithm does not know anything about individual future requests, their bandwidth requirements, or their time of arrival. The time of arrival of individual requests and their classes are entirely unpredictable. Thus, the request sequence is completely online in PEMS.

Traffic Scenario for three users is as follows.

- The volume of each individual demand is fixed at 500kb per second,
- Simulations of our two algorithms (LBWDP and PEMS) are executed for 50 seconds respectively,
- The source and destination pairs are LSR0-LSR9, LSR4-LSR7 and LSR8-LSR5. These pairs are selected randomly among them every demand generation interval,
- One flow generated every two seconds is stopped at a random time,
- PEMS adapts a time based triggering as a triggering policy and updates link state every 3 seconds,
- PEMS and LBWDP are simulated with same traffic scenario.

4.4.2. Performance comparisons

During the simulation, the throughput, the maximum value of the link utilization and end-to-end delay along the LSPs are logged.

4.4.2.1. Throughput

If there is an LSP with low-delay and big residual bandwidth, it can be the candidate path for EF and AF classes. For a period of link state update, this LSP is loaded by the traffic of these two classes. It needs some preemption policy that, if the traffic of two classes tries to send onto one LSP, the priority must be given to EF class and AF class takes another LSP. Since we do not have any preemption policy, the congestion cannot be avoided when the load is heavy. Consequently PEMS has some lost packets when load is relatively high.

Algorithm	LBWDP	PEMS
Lost packets	0.00 %	6.70 %

To estimate the throughput, the three destination routers record the number of received packets. The received packets of EF and AF traffics in PEMS are more than those of LBWDP but the cost is the deterioration of BE traffic.

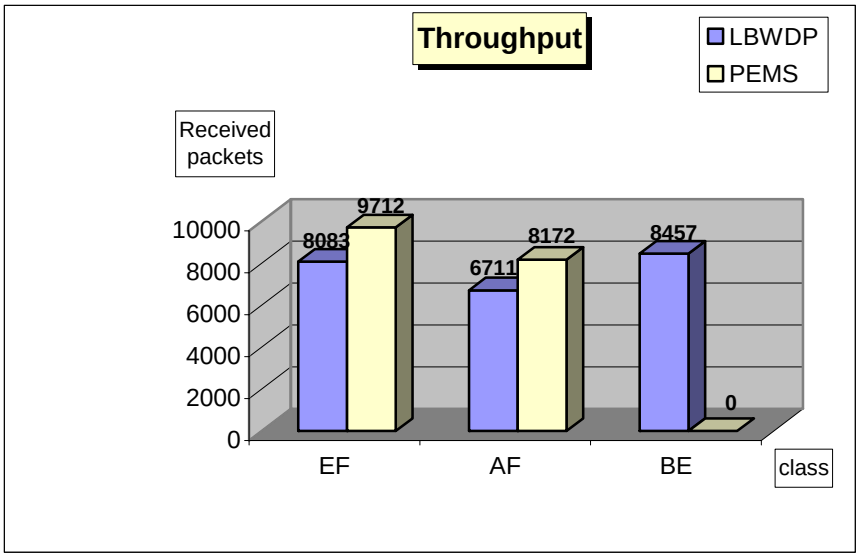


Fig 4.11 Throughput comparison between LBWDP and PEMS

4.4.2.2. Rejection rate

Both algorithms, LBWDP and PEMS, can reject the request when there is no path having the sufficient capacity to assign the current demand. Since they use multipath routing they have few possibility of rejection. Our simulation has no rejection for the two algorithms.

Algorithm	LBWDP	PEMS
Rejection rate	0 %	0 %

4.4.2.3. Delay

The next metric compared is the delay encountered by the request flows. The delay is composed of three components: the transmission, propagation and the queuing delay. The transmit time and link propagation is fixed, however the queuing delay is determined by the load on the link. In other words, for a highly loaded LSP the queuing delay is larger than the value for a lightly loaded LSP. Queuing delay occurs inside the network since routers generally need to store packets for some time before forwarding them on an outbound link. Queuing delay varies with the length of queues in buffers.

- Delay = Transmit Time + Link Propagation + Queue Delay
- Transmit Time = Size/Bandwidth
- Link Propagation = Distance/Speed of Light

LSP_{sd} corresponds to an LSP between ingress s and egress d ($s,d \in N$). l_{ij} is link connecting i and j. A path LSP_{sd} between s and d is defined as a concatenation of l_{sx}, ..., l_{zd} where the nodes x, ... z are arbitrary nodes in the network.

- d_{ij}^{l} : delay incurred on link ij
- d_{sd}^{p} : delay incurred on path LSP_{sd}

The path delay is assigned to be equal to the sum of the delays on the individual links in the path.

$$d_{sd}^{p} = \sum_{(i,j) \in LSP_{sd}} d_{ij}^{l}$$

To estimate delay, queue monitor is used in *ns* simulator. Queue monitoring is implemented by constructing three SnoopQueue objects and one QueueMonitor object. The SnoopQueue objects are linked in around a Queue.

Queue monitoring refers to the capability of tracking the dynamics of packets at a queue. Several classes are used in supporting queue monitoring. When a packet arrives at a link where queue monitoring is enabled, it generally passes through a SnoopQueue object when it arrives and leaves (or is dropped). These objects contain a reference to a QueueMonitor object.

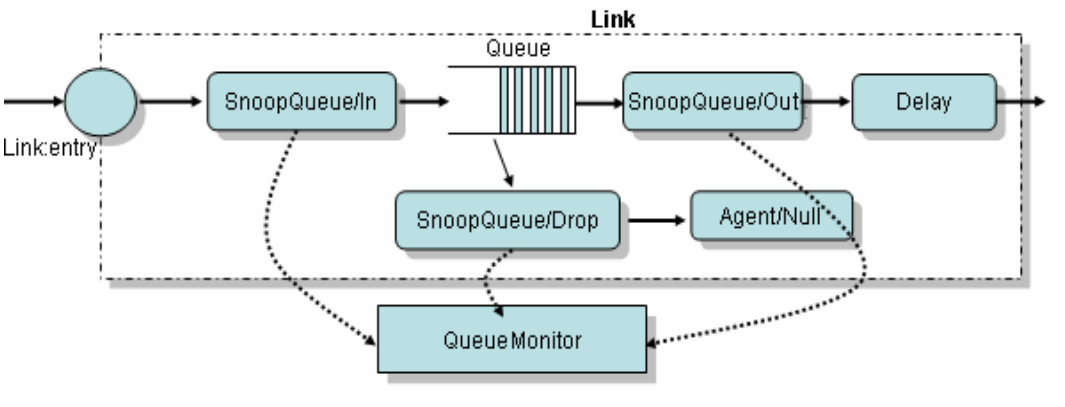


Fig 4. 12 A queueMonitor and supporting objects in *ns-2*

Fig.4.13 is simulation result with topology 4.10. Each class' delay experienced with LBWDP and PEMS are logged. It shows that LBWDP gets the almost same results for all classes indifferently while PEMS can differentiate the measured delay according to the class. With PEMS, EF class has the best performance in delay to guarantee the quality of service. Of course, in this situation the delay experienced by BE traffic is greater than with LBWDP.

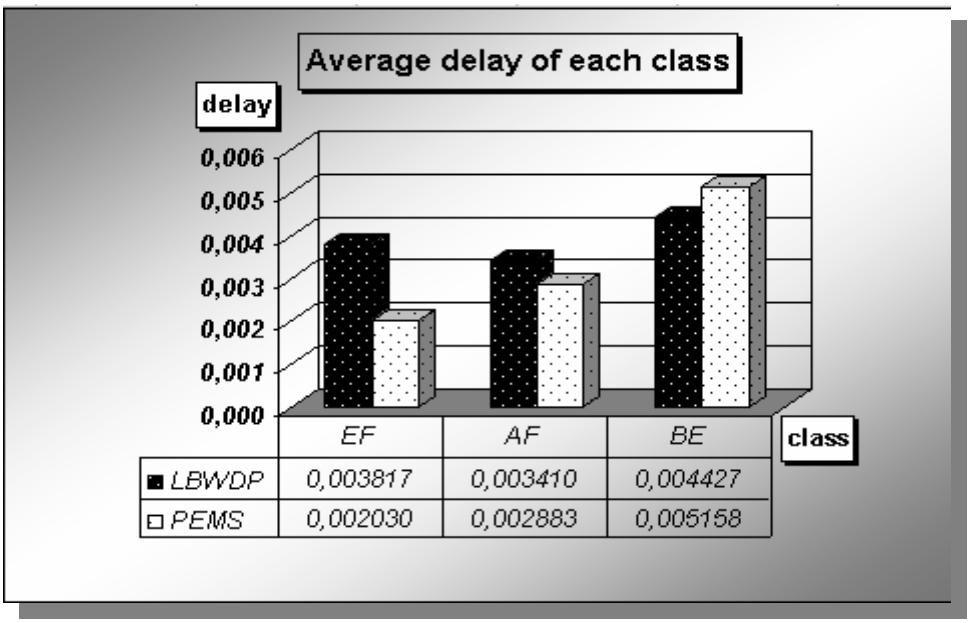


Fig 4. 13 Simulation result of each class' Delay

4.4.2.4. Link utilization

Algorithm	LBWDP	PEMS
Maximum link utilization	78.80 %	96.72 %
Average link Utilization	11.84 %	26.00 %

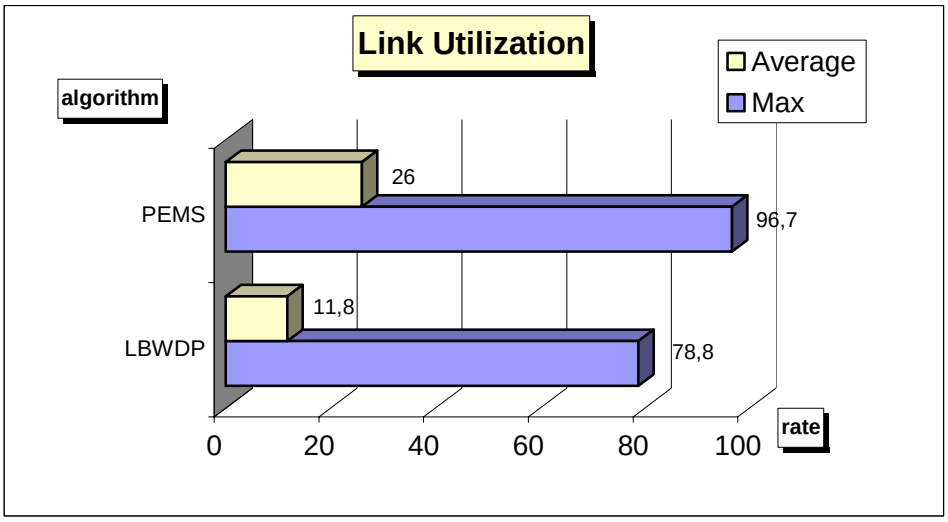


Fig 4. 14 Simulation result of link utilization

LBWDP can balance the load better than PEMS because it does not concern a specific class but PEMS must submit to deterioration of load balancing, not high, at the expense of service differentiation.

4.5. Conclusion

Service differentiation by DS-TE mechanism has certain advantages for a commercial operator since the quality of all accepted traffic is satisfactory and susceptible to charging.

In this chapter, we propose new DS-TE model, PEMS, to verify that DiffServ aware MPLS traffic engineering is useful to meet the customers' various requirements. PEMS distributes three classes, EF for delay sensitive traffic like VoIP traffic, AF for soft bandwidth and loss guarantee traffic like ftp or e-mail and BE for assuring the minimum quality to best effort service. PEMS aims to minimize the maximum link utilization like LBWDP algorithm basically, and additionally to give different service quality to each class, especially to guarantee the low delay to EF class.

In the pre-processing stage of PEMS model, it selects some good paths of all possible paths in the offline mode by new maximally link disjointed paths searching algorithm with minimum total length and shared links between a pair of nodes. This algorithm uses the characteristics of disjointedness for reducing the congestion possibility and for avoiding the exhaustive searching burden in online mode.

Every link state update time, new candidate paths for each class are calculated based on updated information such as measured delay and residual bandwidth. This stage selects multiple low-delayed paths in the ordered paths set as candidate paths of EF class and selects multiple paths having more residual capacity for AF class and takes the remains for BE class.

When the traffic demand is arrived, it uses PER algorithm to select one LSP to carry current demand. We adapt the parameters with different weights of delay and bandwidth depending on the class of requested traffic when calculate the selection probability.

The simulation result showed that PEMS can differentiate the quality of service according to the class. EF class can send the traffic onto the low-delayed path; as a

result it can obtain better quality in terms of measured delay. And in terms of maximum link utilization of the network, the simulation result shows also that PEMS must submit to deterioration of load balancing, not high, at the expense of service differentiation.

Unfortunately using DS-TE traffic engineering severely increases the processing complexity performed by the network nodes, especially edge routers.

Different service availability guarantees constitute a useful key for price discrimination. Admission control remains experimental although its feasibility in large backbone routers has recently been demonstrated and some small-scale trials have produced excellent results [Benameur, 2004]. [Fredj, 2001] presents that Measurement-based implementation is essential to ensure the necessary reactivity and to avoid the well known problems of scalability.

PEMS can be extended for more traffic classes and could work with IntServ model (Integration of services).

Chapter 5:

Conclusions and Future works

We summarize the main contributions that stem from this work before giving the future research directions.

5.1. Summary

In this dissertation, we describe a framework for providing QoS in the Internet. To achieve traffic engineering objective, we focus on the load balancing through the multipath routing in the IP-based MPLS network and DS-TE network. We propose not only the hybrid approach for IP-based MPLS network but also new model PEMS for DS-TE network. PER traffic splitting algorithm is proposed and used as a part of both solutions.

The growth of multimedia applications over wide area networks has increased research interest in Quality of Service. Just increasing the amount of resources such as available bandwidth to avoid congestion does not provide proper resource utilization. Traffic Engineering is concerned with the performance optimization of operational networks. Its main objective is to reduce congestion hot spots and improve resource utilization through carefully managing the traffic distribution inside a network. MPLS allows the implementation of evolved TE mechanisms by providing the possibility of establishing source routed paths.

We set the TE objective to minimize the maximum link utilization which is the best criterion to guarantee an average QoS since this allows reducing the network congestions, packet delay and so on, as a result increasing the reliability and availability

of routers. Multipath routing is the one of the load balancing mechanisms in which the total load from a source to a destination is spatially distributed over several paths. We have established the general multipath routing scheme having two stages: *candidate paths selection and traffic splitting among them.* Following this assessment, four recent multipath algorithms based on MPLS have been studied; WDP, MATE, multipath-AIMD and LDM, with regard to scalability and stability. Each multipath routing algorithm in the literature was declared very efficient by its authors but generally with respect to restricted conditions. In this context, instead of developing a new routing algorithm, the hybrid approach which one step of one algorithm combines with another step of different algorithm is possible to improve its performance than original ones. And naturally the new proposition only for one stage, not the whole routing algorithm, is possible to be used as one modular part of the hybrid algorithms like PER.

Since the optimization objective of this study is to minimize the maximum link utilization in the network, the algorithms which use link utilization rate as a cost metric, WDP and LDM, are selected for our hybrid approaches. We divide these two algorithms into proper steps. And additionally, new traffic splitting method is proposed using the prediction of effective repartition, called PER, as one modular for second stage. At the period between link state update time, routing algorithm calculates its path using information of the time before. When the number of shared links between LSPs increases and when the traffic to send also increases, then the difference between the information kept and the real state increases too. When the difference between the local estimation of residual bandwidth and its real value given by next link state update is high, congestion probability is increased because of the inaccurate information. PER algorithm reduces this inaccuracy. When traffic volume is high, PER proves its efficiency.

With these modular parts, we have proposed the combined three load balancing algorithms for multipath QoS. Three hybrids' simulation results showed that they are satisfiable and particularly the third one, LBWDP's load through the network was best balanced so that the network throughput is improved. LBWDP algorithm is more efficient owing to its second part PER through the simulation especially when traffic

volume is high. We can also propose the possibility of multi-model approach which performs different algorithms corresponding to the load as a future work.

MPLS offers many advantages to service providers. In order to support today's various kinds of applications, the system needs to guarantee the Quality of service. However, MPLS is incapable of providing differentiated service levels in a single flow. Hence MPLS and DiffServ seem to be a perfect match and if they can be combined in such a way to utilize each technology's strong points and counter the others' weaknesses, it can lead to a symbiotic association that can make the goal of end to end QoS feasible. DiffServ-aware Traffic Engineering mechanisms operate on the basis of different Diffserv classes of traffic to improve network performance and extend the base capabilities of TE to allow route computation and admission control to be performed separately for different classes of service.

We propose an approach for providing differentiated services in IP-based MPLS network, called PEMS, to verify that DiffServ aware MPLS traffic engineering is useful to meet the customers' various requirements. Three classes of services are provided, EF for delay sensitive traffic like VoIP traffic, AF for soft bandwidth and loss guarantee traffic like ftp or e-mail and BE for assuring the minimum quality to best effort service. PEMS aims basically to minimize the maximum link utilization over the network like LBWDP algorithm, and additionally to give different service quality to each class, especially to guarantee the low delay to EF class.

PEMS has three stages; pre-processing, candidate paths selection for each class and traffic distribution. It is what pre-processing stage is appended to general multipath routing scheme proposed in section 2.

In the pre-processing phase, it selects some good paths of all possible paths in the offline mode by new maximally link disjointed paths searching algorithm between a pair of nodes. This algorithm uses the characteristics of disjointedness for reducing the congestion possibility and of path length. This stage is prerequisite work to pick up some good paths for avoiding the exhaustive searching burden in the online mode. It is true that its complexity is high but it is not important because it is done in the offline mode and infrequently.

Every link state update time, new candidate paths for each class are calculated based on updated network state information such as measured delay and residual bandwidth. This stage selects multiple low-delayed paths in the ordered paths set as candidate paths of EF class and selects multiple paths having more residual capacity for the AF class and BE class takes the remaining.

When the traffic demand is arrived, it uses PER algorithm to select one LSP to assign. We adapt the parameters with different weights of delay and bandwidth depending on the class of requested traffic when calculate the selection ratio.

The simulation results show that PEMS can differentiate the quality of service according to the class. EF class can send the traffic onto the low-delayed path; as a result it can obtain better quality in terms of measured delay. And in terms of maximum link utilization of the network, PEMS must submit to deterioration of load balancing, not high, at the expense of service differentiation. This simulation uses a single topology, a single signalling protocol, and a single type of traffic, among other simplifications. The simulator itself should be improved taking into account complex interactions as found in the Internet today.

5.2. Future works

Providing QoS in the Internet is an important but difficult task. In this study, we propose some efficient routing algorithms. These propositions such as three hybrids and PEMS have been individually evaluated through simulation. But it is still theoretical and the simulation model and experiments are small scale with the sole aim of providing a corroborative demonstration for the objective made for this dissertation.

There are still many works remaining in the progress of end-to-end QoS management for multi-service in IP networks. We will address the following issues in future works:

1. *Amelioration of propositions* - In PEMS, candidate path's splitting ratio calculation stage can be ameliorated by adapting dynamically the parameters of

the splitting ratio equation depending on the network state. And also PEMS can be extended for more traffic classes and could work with IntServ model (Integration of services).

2. *Estimation in the testbed* - Our proposed algorithms must be estimated in the physical testbed, with Next Generation Internet routers, equipped with DiffServ-capable routers and switches especially for PEMS. It must be done using more realistic traffic models for the Internet.
3. *Extension to multi-domains* - The work should include the inter-DS domain since this thesis only discusses the intra domain. Further researches would include extending the managing area to a large network, composed of several domains. This is needed to achieve end-to-end QoS for applications. The large and complex network challenge results from a large number of autonomous systems in the end-to-end path.

References

[Abdellatif, 2002] S. Abdellatif, G. Juanole, "*Cadre basé sur les réseaux de Petri pour l'analyse de la Qualité de Service*", Technical Rapport LAAS N° 02205, LAAS-CNRS, 2002.

[Abdellatif, 2005] S. Abdellatif, G. Juanole, "*Proposition et évaluation par réseaux de Petri stochastiques d'une mise en oeuvre du traitement assuré DIFFSERV "*, 11ème Colloque Francophone sur l'Ingénierie des Protocoles (CFIP'2005) pp.41-49, Bordeaux, France, Avril 2005.

[Ahn, 2001] G. Ahn, W. Chun, "*Design and Implementation of MPLS Network Simulator (MNS) Supporting QoS*", 15th International Conference on Information Networking (ICOIN'01), Beppu City, Japan, Feb. 2001.

[Akar, 2003] N. Akar, I. Hokelek, M. Atik, E. Karasan, "*A Reordering-free Multipath Traffic Engineering Architecture for DiffServ-MPLS Networks*", 3rd IEEE Workshop on IP Operations & Management (IPOM2003), Kansas City, Missouri, Oct. 2003

[Auriol, 2004] G. Auriol, C. Chassot, M. Diaz, "*Architecture de communication à gestion automatique de la QdS en environment IP à services différenciés*", Rapport LAAS No 02469, Technique et Science Informatiques, Vol.23, N°9, 2004.

[Bai, 2003] X. Bai, "*Investigation and Analysis of Optimal Multi-class Routing Algorithms for IP networks*", Master thesis of Helsinki university of technology, Finland, April 2003.

[Benameur, 2004] N. Benameur, A. Kortebi, S. Oueslati, J.W. Roberts, "*Selective service protection in overload; differentiated services or per-flow admission control?*", 2004

[Bertsekas, 1992] D. Bertsekas, R. Gallager, "*Data networks*", Prentice Hall, 1992.

[Bohacek, 2003] S. Bohacek, J. Hespanha, J. Lee, C. Lim, K. Obraczka, "*TCP-PR: TCP for Persistent Packet Reordering*", In Proc. of the IEEE 23rd

Int. Conf. on Distributed Computing Systems, pp 222-231, May 2003.

[Bonald, 2003] T. Bonald, J.W. Roberts, "*Congestion at flow level and the impact of user behaviour*", Computer Networks Vol 42, pp 521-536, 2003

[Caterina, 2004] S. Caterina, A. Tricha, C. Jaudelice, U. George, "*TEAM: A traffic Engineering Automated Manager for DiffServ-Based MPLS Network*", IEEE Communication Magazine pp.134-145, Oct. 2004.

[Chassot, 2006] C. Chassot, A. Lozes, F. Racaru, G. Auriol, M. Diaz, "*A user-based approach for the choice of the IP services in the multi domains DiffServ Internet*", International Workshop on Service Oriented Architectures in Converging Network Environments (SOCNE 2006), Vienna, Austria, April 2006.

[Chen, 1999] J. Chen, "*New approaches to Routing for Large-Scale Data Networks*", Thesis of Doctor, RICE Univ., Houston, Texas, June, 1999.

[Chpenst, 2001] A. Chpenst and T. Curran, "*Optimization of QoS traffic with a number of constraints in DiffServ/MPLS networks*", http://telecoms. eeng.dcu.ie/symposium/papers/D1.pdf

[Cormen, 1990] T. H. Cormen, C. E. Leiserson, R. L. Rivest, "*Introduction to Algorithms*", MIT Press and McGraw Hill, 1990

[CPLEX] ILOG CPLEX 9.0 Getting Started, http://esra.univ-paris1.fr/cplex/pdf/gscplex.pdf.

[Daniel, 1999] Daniel O. A, "*MPLS and Traffic Engineering in IP networks*", IEEE Communications Magazine, pp 42-47, December 1999.

[Dijkstra, 1959] E. W. Dijkstra. "*A note on two problems in connection with graphs*", Numerische Mathematik, 1: pp83-89, 1959.

[Dogar, 2005] F. Dogar, Z. Uzmi, S. Baqai, "*CAIP: A Restoration Routing Architecture for DiffServ Aware MPLS Traffic Engineering*" IEEE Globecom, 2005

[Eleni, 2003] M. Eleni, C. Charalampos, C. Panos, D. Takis, G. Danny, T. Panos, P. George, G. David, "*Admission Control for providing QoS in*

DiffServ IP networks: The TEQUILA approach", IEEE Communications magazine pp.38-44, Jan, 2003.

[Elwalid, 2001] A. Elwalid, C. Jin, S. Low, and I. Widjaja, "*MATE: MPLS Adaptive Traffic Engineering*", INFOCOM'2001, pp. 1300-1309, Alaska, 2001.

[Eppstein, 1998] D. Eppstein, "*Finding the K shortest Paths*", SIAM J. Computing, 28(2):652-673, 1998

[EuQoS] EuQoS project, End-to-end Quality of Service support over heterogeneous networks, http://www.euqos.org

[Faucheur, 2002] F. Le Faucheur, "*Considerations on bandwidth constraint models for DS-TE "*, IETF Internet Draft, work in progress, June 2002.

[Fineberg, 2003] V. Fineberg, "*QoS Support in MPLS Networks: MPLS/Frame Relay Alliance White Paper*", MPLS FORUM, May 2003,

[Fortz, 2000] B. Fortz and M. Thorup, "*Internet Traffic Engineering by Optimizing OSPF Weights*", INFOCOM2000, 2000.

[Fredj, 2001a] S. Fredj, S. Oueslati-Boulahia, J. W. Roberts, "*Measurement-based Admission Control for Elastic Traffic*", ITC 17, Salvador, Brazil, December 2001.

[Fredj, 2001b] S. Fredj, T. Bonald, A. Proutiere, G. Régnié, J. W. Roberts, "*Statistical Bandwidth Sharing: A Study of Congestion at Flow Level*", ACM SIGCOMM 2001, pp.111–122, 2001

[Guerin, 1997] R. Guerin, A. Orda., D. Williams, "*Qos Routing Mechanisms and OSPF Extensions*", In Proc. of the Global Internet Miniconference, Phoenix, USA, 1997.

[Hedrick, 1991] Charles L. Hedrick, "*An Introduction to IGRP*", CISCO White paper, August 1991.

[Ji-Feng, 2003] C. Ji-Feng, H. Zuo-Po, L. Chi-Wen, H. Wen-Shyang, S. Ce-Kuen, "*Supporting End-to-End QoS in DiffServ/MPLS Networks*", IEEE, 2003.

[Kim, 2004a] Y. Kim, H. Kim, "*MPLS 망에서의 QoS 보장형 서비스 제공을 위한 DiffServ-aware-MPLS 연결관리 기능연구*", KNOM Review, Vol.6, No.2, pp.69-79, Korean Institute of Communication Sciences (KICS), Feb.2004. [Kim, 2004a]

[Kim, 2004b] Y. Kim, "*DoumiMan (DiffServ-over-universal-MPLS Internet Manager) for Guaranteed QoS Provisioning in Next Generation Internet*," IEEE COMSOC, NOMS2004 (Network Operations and Management Symposium), COEX, Seoul, April 2004.

[Kim, 2005a] Y. Kim, H. Kim, H. Shin, "*Session and Connection Management for QoS-guaranteed Multimedia Service Provisioning on IP/MPLS Networks*" Springer Verlag, ICCSA (International Conference on Computational Science and its Applications) 2005, Singapore, May, 2005.

[Kim, 2005b] Y. Kim, H. Kim, "*Per-Class-type Virtual Networking for QoS-guaranteed DiffServ Provisioning on IP/MPLS Networks*", ICC2005.

[Lee, 2002] Y. Lee, Y. Seok, Y. Choi, C. Kim, "*A constraind multipath traffic engineering scheme for MPLS networks*," IEEE ICC2002, April, 2002.

[Lee, 2005a] K. Lee, A. Noce, A. Toguyéni, A. Rahmani, "*Comparison of multipath algorithms for load balancing in a MPLS Network*", ICOIN'05, Published by Springer as Lecture Notes in Computer Science (LNCS), Vol. 3391, pp.463~470, February 2005.

[Lee, 2005b] K. Lee, A. Toguyéni, A. Rahmani, "*Stable load balancing algorithm in MPLS network*", IMACCA'05, pp.47-51, Marseille, France, Oct. 2005

[Lee, 2006a] K. Lee, A. Toguyéni, A. Rahmani, "*Hybrid multipath routing algorithms for Load balancing in MPLS based IP network*", AINA'2006, pp.165-170, Vienna, Austria, April 2006

[Lee, 2006b] K. Lee, A. Toguyéni, A. Rahmani, "*Comparison of Multipath Algorithms for Load Balancing in a MPLS network*", EITA'06 (Encyclopedia of Internet Technologies and Applications), 2006

[Lee, 2006c] K. Lee, A. Toguyéni, A. Rahmani, "*Periodic Multi-Step routing algorithm for DS-TE: PEMS*", International e-Conference on Computer Science 2006 (IeCCS'2006), July 2006

[Lepage, 2006] F. Lepage, T. Divoux, F. Michaut, "*Adaptation of control parameters based on QoS monitoring*", 17th International Symposium on Mathematical Theory of Networks and Systems, Kyoto, Japan, July, 2006.

[Leung, 2000] K. Leung, "*Multipath routing In Communication network*", Ph.D. thesis, Univ.of Southern California, Dec 2000.

[MATLAB] The language of technical computing, http://www.mathworks.com/products/matlab/.

[Minei, 2004] I. Minei, "MPLS DiffServ-aware Traffic Engineering", White paper of Juniper networks, 2004.

[MNS] The MPLS Network Simulator version 2 (*ns-2*), http://flower.ce.cnu.ac.kr/~fog1/mns/

[Moh, 2001] M. Moh, B. Wei and J. H. Zhu, "*Supporting differentiated services with per-class traffic engineering in MPLS*", International Conference on Computer Communications and Networks 2001, pp.354-360, 2001

[NS2] The Network Simulator *ns-2*, http://www.isi.edu/nsnam/ns/

[Oki, 1995] E. Oki and N. Yamanaka, "*A recursive matrix-calculation method for disjoint path search with hop link number constrints*", IEICE Trans.Commun., vol.E78-B, no5, pp.769-774, 1995.

[Oliveira, 2004] J. C. de Oliveira, C. Scoglio, I. Akyildiz, G. Uhl, "*Traffic Preemption Policies for DiffServ-Aware Traffic Engineering to Minimize Rerouting in MPLS Networks*" IEEE/ACM Transactions on Networking, vol10,No.4, pp.733-745, August 2004.

[Owezarski, 2002] P. Owezarski, "*Using Congestion Control aggressiveness for Service Differentiation in the Internet*", Rapport LAAS n° 02208, April 2002

[Owezarski, 2003] P. Owezarski, N. Larrieu, "*Coherent Charging of Differentiated Services in the Internet Depending on Congestion Control Aggressiveness*", Rapport LAAS N°02266, Computer Communications, Issue 13, Vol.26, pp.1445-1456, Août 2003

[Padhye, 1998] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, "*Modeling TCP Throughput: A Simple Model and its Empirical Validation*", SIGCOMM'98, 1998

[Palakurthi, 2001] H. Palakurthi, "*Study of Multipath Routing for QoS Provisioning*", Introduction to research, Oct. 2001.

[Park, 2002] I. Park, J. Kim, Y. Lee, Y. Choi, "*Stable Load Control in Multipath Packet Forwarding*", Journal of KISS, vol 29, no 2, Apr. 2002.

[Pasquale, 2004] L. Pasquale, "*DiffServ Traffic Management with MPLS*", Master thesis, University of Pisa, Italy, Dec.2004.

[Paulo, 2005] P. Paulo, L. Pasquale, C. Augusto, "*DiffServ Traffic Management with MPLS*, 5th Conference on Telecommunications (ConfTele'2005), Tomar, Portugal, April 2005.

[Peerapon, 2003] S. Peerapon, B. Sujata, T. David, "A Survey of Adaptive Bandwidth Control Algorithms," IEEE Communications Surveys and Tutorials, Vol.5, no.1, June 2003.

[Pu, 2002] J. Pu, E. Manning, G. C. Shoja, "Reliable Routing in MPLS Networks", Proc IASTED International Conference of Communications and Computer networks (CCN 2002), 2002.

[Raghava, 2001] S. Raghavan, "*An MPLS-based Quality of Service Architecture for Heterogeneous Networks*", Master Thesis of Virginia Polytechnic Institute and State University, Nov.2001.

[Rao, 1998] N. Rao, S. Batsell, "QoS Routing Via Multiple Paths Using Bandwidth Reservation", IEEE INFOCOM'98, March 1998.

[RFC1058] C. Hedrick, “Routing Information Protocol”, RFC1058, Jun.1988, IETF.

[RFC1105] K. Lougheed, Y. Rekhter, “Border Gateway Protocol (BGP)”, RFC1105, June 1989, IETF.

[RFC1633] R. Braden, D. Clark, S. Shenker, “*Integrated Service in the Internet Architecture: an Overview*”, RFC1633, Jun. 1994, IETF

[RFC2205] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin, “*Resource ReSerVation Protocol (RSVP)*”, RFC2205, Sept 1997, IETF

[RFC2212] S. Shenker, C. Partridge, and R. Guerin, “*Sprcification of Guaranteed Quality of Service*”, RFC 2212, Sept., 1997, IETF.

[RFC2215] S. Shenker, J. Wroclawski, “*General Characterization Parameters for Integrated Service Network Elements*”, RFC 2215, Sept., 1997, IETF.

[RFC2328] J. Moy, “*OSPF version 2*”, RFC2328, Apr 1998, IETF

[RFC2386] E. Crawley, R. Nair, B. Rajagopalan and H. Sandick, “*A Framework for QoS-based Routing in the Internet*”, RFC 2386, Aug. 1998, IETF.

[RFC2430] T. Li, Y. Rekhter, “*A Provider Architecture for Differentiated Services and Traffic Engineering (PASTE)*”, RFC 2430, Oct.1998, IETF.

[RFC2474] K. Nichols, S. Blake, F. Baker, and D. Black, “*Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*”, RFC 2474, Dec., 1998, IETF.

[RFC2475] D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, “*An Architecture for Differentiated Service*”, RFC2475, Dec 1998, IETF

[RFC2702] D. Awduche, J. Malcolm, J. Agogbua, M. O’Dell, J. McManus, “*Requirement of Traffic Engineering over MPLS*”, RFC2702, Sept 1999, IETF

[RFC2748] D. Durham, J. Boyle, R. Cohen, R. Rajan, and A. Sastry, "*The COPS(Common Open Policy Service) Protocol*", RFC 2748, Jan. 2000, IETF.

[RFC2917] K. Muthukrishnan, A. Malis, "Core MPLS IP VPN Architecture", RFC 2917, September 2000.

[RFC2992] C. Hopps, "*Analysis of an Equal-Cost Multi-Path Algorithm*", RFC2992, Nov 2000, IETF

[RFC3031] E. Rosen, A.Viswanathan, R.Callon, "*Multiprotocol Label Switching Architecture*", RFC3031, Jan. 2001, IETF.

[RFC3035] L. Andersson, P. Doolan, N. Feldman, A. Fredette, B. Thomas, "LDP Specification", RFC3035, Jan. 2001, IETF.

[RFC3270] F. Le Faucheur, L. Wu, B. Davie, S. Davari, P. Vaananen, R. Krishnan, P. Cheval, J. Heinanen, "*MPLS Support of Differentiated Services*", RFC 3270, May, 2002, IETF.

[RFC3564] F. Le Faucheur, W. Lai, "*Requirements for Support of Differentiated Services-aware MPLS Traffic Engineering*", RFC 3564, July, 2003, IETF.

[RFC4126] J. Ash, "*Max Allocation with Reservation Bandwidth Constraints model for DiffServ-aware MPLS Traffic Engineering & Performance Comparisons*", RFC 4126, June.2005, IETF.

[RFC4128] W. Lai, "*Bandwidth Constraints Models for Differentiated Services (DiffServ)-aware MPLS Traffic Engineering: Performance Evaluation*", RFC4128, June 2005, IETF.

[Saad, 2001] T. Saad, T. Yang, D. Makrakis, V. Groza, "*DiffServ-enabled Adaptive Traffic Engineering over MPLS*", ICII2001, Vol.2 pp 128-133, 2001.

[Salama, 1997] H. Salama, D. Reeves, Y. Viniotis, "*A Distributed Algorithm for Delay-Constrained Unicast Routing*", IEEE INFOCOM '97, 1997.

[Sergio, 2004] B. Sergio, "*Techniques d'Optimisation pour le Dimensionnement et la Reconfiguration des Réseaux MPLS*", Ph.D Thesis, Ecole

Nationale supérieure des télécommunications, Paris, France, April, 2004

[Shen, 2002] J. Shen, J. Shi, J. Crowcroft, "*Proactive Multi-path Routing*", Third International Workshop on Quality of Future Internet Services (QofIS 2002), Lecture Notes in Computer Science 2511 Springer, pp. 145 - 156, Zurich, Switzerland, October 2002.

[Shigang, 1999] C. Shigang, "*Routing Support for Providing Guaranteed End-to-End Quality-of-Service*", Ph.D. thesis, UIUC, 207 pages, 1999.

[Skiena, 1990] S. Skiena, "*Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica"*, Reading, MA: Addison-Wesley, 1990.

[Song, 2003] J. Song, S. Kim, M. Lee, "*Dynamic Load Distribution in MPLS Networks*", LNCS Vol. 2662, pp.989-999, 2003.

[Srihari, 2001] N. Srihari, Z. Zhi-Li, "*On Selection of Paths for Multipath Routing*", In Proc. IWQoS'01, Karlsruhe, Germany, 2001.

[Srinivas, 2001] V. Srinivas, "*Multipath routing machanisms for traffic engineering and quality of service in the Internet*", Ph.D thesis, University of California, Santa Cruz, March, 2001

[Susitaival, 2002] R. Susitaival, "*Load balancing by MPLS in differentiated services networks*", Master thesis of Helsinki university of Technology, Finland, Sept. 2002.

[T'kindt, 2002] V. T'kindt, B. Jean-Charles, "*Multicriteria Scheduling: Theory, Models and Algorithms*", Springer, 300 pages, 2002.

[Tanenbaum, 1996] A. Tanenbaum, "*Computer Networks*", Third edition, Prentice Hall PTR, 1996.

[Vogel, 1995] A. Vogel, B. Kerherve, G. Bochmann, J. Gecsei, "*Distributed Multimedia and QOS: a Survey*", IEEE Multimedia, Vol.2, No.2, pp.10-19, Summer 1995.

[Vutukury, 1999] S. Vutukury and J. J. Garcia-Luna-Aceves, "*A Distributed Algorithm for Multipath Computation*", Globecom'99, pp.1689-1693, 1999.

[Wang, 1996] Z. Wang, J. Crowcroft, "*QoS Routing for Supporting Multimedia Applications*", IEEE Journal of Selected Areas in Communications 14, pp1228-1234, 1996.

[Wang, 1999] Y. Wang, Z. Wang, "Explicit Routing Algorithms for Internet Traffic Engineering," In ICCCN'99 Proceedings pp.582-588, September 1999.

[Wang, 2002] J. Wang, S. Patek, H. Wang, J. Liebeherr, **"***Traffic Engineering with AIMD in MPLS Networks*", LNCS, May, 2002.

[Xiao, 2000] X. Xiao, "*Providing Quality of service in the Internet*", PhD thesis, Michigan State University, Jan. 2000

[Yang, 2003] Y. Yang, L. Zhang, J. K. Muppala, S. T. Chanson, "*Bandwidth-delay constrained routing algorithms*", Computer networks 42, pp.503-520, 2003

[Zhang, 2001] W. Zhang, "*Internet QoS; Architectures and Mechanisms for Quality of Service*", Morgan Kaufmann Publishers, 2001

[Zhang, 2002] P. Zhang, R. Kantola, S. Aalto, "*QoS Routing for DiffServ Networks; Issues and Solutions*", Technical Report, Oct. 2002

Annexes

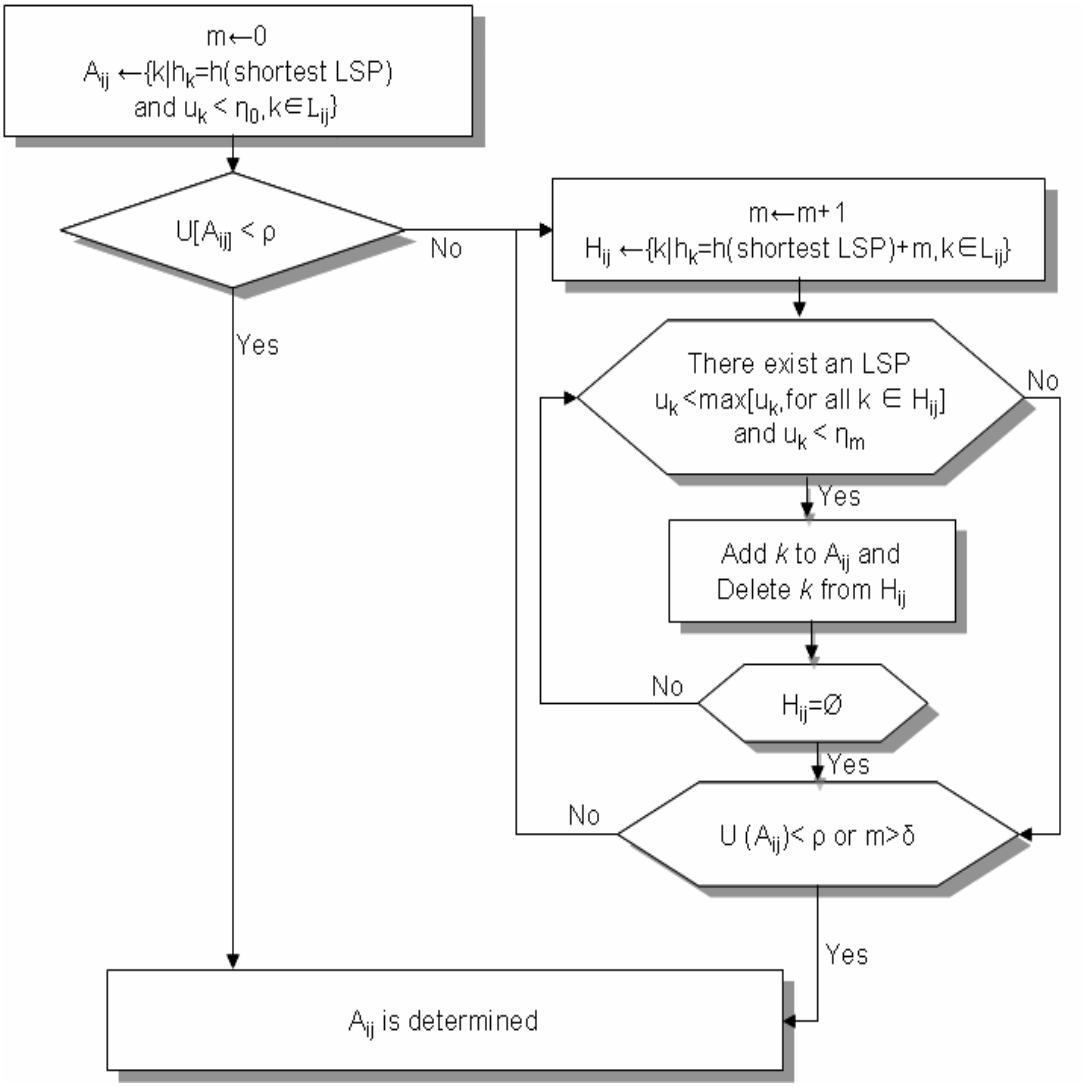
Annex A : Improvement of original WDP algorithm

```
PROCEDURE SELECT(η ,ψ)
    W+ =0;
    For each path r in Rσ
        For each link l in r
            cl = cl - ( 1 - br ) vr
    For each path r in R̂σ \ Rσ
        If |Rσ| < η
            Wr = WIDTH(Rσ U r)
        Else
            Wr = WIDTH(Rσ U r \ NARROWEST(Rσ U r))
        Endif
        If (Wr > W+) then W+ = Wr ; End if
    End for

    W+ = max r∈R̂σ \ Rσ Wr
    If ( W+ > ( 1+ ψ)WIDTH(Rσ))
        R+ =Ø
        For each path r in R̂σ \ Rσ
            If (Wr = W+) then R+ = R+ U r ; End if
        End for
        d+ = ∞
        For each path r in R̂σ \ Rσ
            If (d+ > dr) then d+ = dr ; End if
        r+ = Ø
        For each path r in R+
            If (d+ = dr) then r+ = r+ U r ; End if
        End for
        For each path r in r+
            r- = NARROWEST (Rσ U r)
            If (|Rσ| = η or WIDTH ( Rσ U r+ \ r-) = W+)
                Rσ = (Rσ U r+) \ r-
            Else
                Rσ= Rσ U r+
            End if
    Else
        r- = NARROWEST(Rσ)
```

```
            If WIDTH (Rσ\ r-)=WIDTH(Rσ)
                  Rσ = Rσ U r-
            End if
      End if
END PROCEDURE
```

Annex B : Path selection part of LDM algorithm



Titre : **Modèle global pour la Qualité de Service dans les réseaux de FAI : intégration de DiffServ et de l'ingénierie de trafic basée sur MPLS**

Le routage multi-chemins est une technique qui permet l'équilibrage de la charge en multiplexant les flux vers une destination sur plusieurs chemins. Nous proposons une approche de routage multi-chemins qui, peut être schématisée en deux étapes : l'étape de choix des chemins candidats et l'étape de distribution du trafic sur un sous-ensemble de ces chemins.
Dans ce travail, nous avons commencé par effectuer une étude comparative d'un point scalabilité et stabilité, de plusieurs algorithmes de routage multi-chemins, basés sur MPLS. Cela nous a permis de retenir WDP pour la sélection des chemins candidats et LDM pour la distribution des demandes de trafic reçues par un routeur entrant du réseau d'un FAI. Nous proposons dans ce travail PER, un algorithme qui est une amélioration de l'algorithme de distribution de LDM. Ces différents algorithmes nous ont permis de réaliser plusieurs algorithmes « hybrides » dont LBWDP (Load Balancing over Widest Disjoint Paths) qui par simulation a été prouvé comme étant un algorithme plus performant que des modèles comme LDM ou MATE.
Pour une meilleure garantie de QdS nous avons cherché à intégrer la différentiation de service (DiffServ) avec notre technique d'ingénierie de trafic (DiffServ-aware MPLS Traffic Engineering : DS-TE). Nous proposons PEMS (PEriodic Multi-Step algorithm for DS-TE network) comme un modèle de DS-TE pour différencier la qualité du service selon la classe du trafic. Dans ce cadre, nous proposons un nouvel algorithme de sélection de chemins candidats en fonctions des critères préférentiels de chaque classe de trafic. L'utilisation de PER permet ensuite de distribuer en fonction de critères dynamiques les demandes reçues sur les meilleurs chemins de chaque classe. Par simulation à l'aide de *ns-2*, nous avons montré que PEMS répartie moins bien la charge que LBWDP mais que les classes EF et AF ont une qualité de service meilleure que dans le cas de LBWDP.

Mots clés : L'ingénierie de trafic, Le routage multi-chemins, la Qulaité de Service, Intégration de Diffserv et de l'ingénierie de trafic basée sur MPLS, L'équilibrage de la charge, DiffServ, Dynamic routing algorithm

Title ; **Global QoS model in the ISP networks: DiffServ aware MPLS Traffic Engineering**

Traffic Engineering is the one of the methods for improving the Quality of Service in the network like Internet. Multipath routing is a mechanism for load balancing in which the total load is spatially distributed over several paths. It can reduce congestion probability given by routing into best one path in the current Internet. We focus on MPLS network, which is usable in the ISP network. LSPs of MPLS provide path controllability by explicit route in the connectionless IP network.
Multipath routing scheme is established with two steps, selection of multiple candidate paths and traffic splitting among these selected paths. We compare some recent multipath routing algorithms based on MPLS in the point of scalability and stability. Of them, we pick up WDP and LDM for our hybrid approach which combines cross-steps of different algorithms. PER algorithm is proposed in the improvement of LDM's traffic splitting step. Using LDM, WDP and PER, three hybrid QoS algorithms by each step combination are made. Their performances are proved by simulation that they, especially LBWDP (Load Balancing over Widest Disjoint Paths), are more effective than LDM or MATE.
For better QoS guarantee, we integrate DiffServ with our traffic engineering technique (DiffServ-aware MPLS Traffic Engineering: DS-TE). PEMS (PEriodic Multi-Step algorithm in DS-TE network) is proposed as one DS-TE model to differentiate the quality of service according to the class. For PEMS algorithm, we propose new algorithm to select candidate paths by proper metric of each traffic class. PER algorithm distributes the traffic among its own candidate paths by dynamic metric of requested class. Simulation results using *ns-2* showed that PEMS can balance the load less than LBWDP but EF and AF classes can be ensured the better quality of service than LBWDP.

Keywords : Traffic Engineering, Multipath routing, Quality of Service, DiffServ, DS-TE(DiffServ aware MPLS Traffic Egineering), Load-balancing, Dynamic routing algorithm