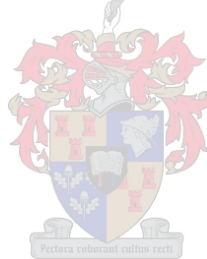


Traffic Engineering Label Switched Paths

Bigomokero Antoine Bagula



Thesis presented in partial fulfilment
of the requirements for the degree of
Master of Science
at the University of Stellenbosch

Supervisor: Prof. A. E. Krzesinski

March 2002

Declaration

I, the undersigned, hereby declare that the work contained in this thesis is my own original work and has not previously in its entirety or in part been submitted at any university for a degree.

Signature:

Date:

Abstract

The Internet is evolving into a commercial platform requiring enhanced protocols and an expanded physical infrastructure allowing a better delivery from IP. Multi-protocol Label Switching (MPLS) is a technology enabling traffic engineering and virtual private network (VPN) provisioning. MPLS achieves traffic engineering by carrying the traffic over virtual connections called Label Switched Paths (LSPs) which are engineered based on QoS requirements such as delay, jitter and packet loss minimization or throughput maximization.

This thesis proposes path finding and traffic distribution methods to be deployed in MPLS networks for traffic engineering LSPs. A flow optimization model based on a pre-planned routing approach separating path finding and traffic distribution is presented. This model is augmented by a threshold routing approach which routes the traffic based on thresholds expressing the maximum load level reached by network links. This routing approach moves the traffic away from threshold-marked links to achieve low-utilized links/paths. The performance and routing capabilities of these methods are evaluated through designed software. A routing architecture implementing a two-layer signalling model for MPLS network is proposed and evaluated through simulation.

Opsomming

Die verandering van die Internet in 'n kommersiële platform met verbeterde protokolle en 'n uitgebreide fisieke infrastruktuur stel die internetprotokol (IP) in staat tot beter lewering. Multi-protokol-etiketstekeling (MPLS), is 'n tegnologie vir die voorsiening van televerkeerbeheer en virtuele privaatnetwerke (VPN). MPLS verskaf televerkeerbeheer deur die verkeer te dra oor virtuele konneksies, wat bekend staan as etiketgeskakelde paaie, waarvan die ontwerp gebaseer is op vereistes vir diensgehalte soos vertraging, ritteling en die minimering van pakketverlies of maksimering van deurvoer.

Hierdie tesis stel nuwe padvind- en verkeerdistribusiemetodes voor wat aangewend word in MPLS-netwerke om etiketgeskakelde paaie te beheer. 'n Model vir vloei-optimering—gebaseer op voorafbeplande roetering wat padvinding en verkeerdistribusie skei—word aangebied. Hierdie model word uitgebrei deur 'n benadering van drempelroetering wat die verkeer roeteer en gebaseer is op drempels wat die maksimum ladingsvlak voorstel wat bereik kan word deur netwerkskakels. Hierdie roeteringsbenadering skuif die verkeer weg van drempelgemerkte skakels en bereik daardeur laaggebruikte skakels/paaie. Die prestasie en roeteringsvaardigheid van hierdie metodes word gevalueer deur selfontwikkelde programmatuur. 'n Argitektuur vir roetering wat 'n dubbellaagseinmodel implementeer vir 'n MPLS-netwerk, word aangebied en gevalueer met simulاسie.

Acknowledgements

First my thanks go to my advisor, Professor Anthony E. Krzesinski who by his sense of responsibility, his insistence on correctness and precision, made this dissertation possible.

I would like also to thank my colleagues of the COE group for their willingness to help i many occasions. My special thanks go to Abriette Senekal and Johan de Kock whose contribution to this work is most valuable.

This work would not be possible without the financial support of TELKOM SA and SIEMENS through the Centre of Excellence in ATM and Broadband Networks (COE).

Many friends, brothers, and all my family have been supportive in many ways since our arrival in Cape Town. I would like to thank all of them.

"Climbing the mountain, step by step, until we reach the top" is the most important principle that my wife Yvette, my children Herman, Fortuna, Nancy and Amani and myself learned during my seemingly never ending student life. I would like to thank deep from my heart Yvette for her support and love and the children for their love and friendship by trying to be the best at school.

Mentioning each person who has contributed to the success of this thesis work is a hard and difficult task. I, therefore, would like to apologize to those who are not explicitly mentioned here.

Contents

Abstract	v
Opsomming	vii
Acknowledgements	ix
1 Introduction	1
1.1 Current Status of IP Networks	1
1.1.1 Internet Protocols	1
1.1.2 QoS Architectures	2
1.1.3 Pricing the Internet	3
1.1.4 IP Traffic Management	3
1.2 Traffic Engineering	4
1.3 Traffic Regulation	5
1.4 Related Work	6
1.4.1 Traffic Engineering	6
1.4.2 Traffic Regulation	7
1.5 Thesis Outline and Main Contributions	10
1.5.1 Key Contributions	10
1.5.2 Thesis Outline	11
2 Traffic Engineering Model	13
2.1 Introduction	13

2.2	A Basic Flow Optimization Model	14
2.2.1	Path Identification	15
2.2.2	Path Selection and Activation	16
2.2.3	Traffic Distribution	18
2.3	Solution Characteristics	25
2.3.1	WTD Optimality	25
2.3.2	FLD Optimality	27
2.3.3	Fault-tolerance	28
2.4	Performance Analysis	29
2.5	Conclusion	35
3	Threshold Routing	37
3.1	Introduction	37
3.2	The Threshold Routing Approach	38
3.2.1	Link Weight Optimization(LWO)	38
3.2.2	Differentiated Routing Regime (DRR)	40
3.3	Performance Measurement	48
3.3.1	Path Characteristics	48
3.3.2	Link Cost Optimization (LCO)	48
3.4	Differentiated Services Models	51
3.4.1	The Path-differentiated Model	51
3.4.2	The Path-multiplexed Model	53
3.5	Performance Analysis	53
3.6	Conclusion	58
4	Proposed Routing Architecture	61
4.1	Introduction	61
4.2	The Proposed Routing Architecture	62

4.2.1	The Key Routing Features	63
4.2.2	The Proposed Network Architecture	64
4.2.3	The Edge-Edge Control Plane	64
4.2.4	The User-User Control Plane	67
4.2.5	The Proposed Edge Router Architecture	67
4.3	Scalability Issues	69
4.3.1	Decentralized Routing	69
4.3.2	Multi-path Routing	70
4.3.3	Packet Reordering	71
4.3.4	Traffic Measurement	71
4.3.5	Routing Updates	72
4.4	Implementation Strategy	74
4.4.1	Traffic Measurement	74
4.4.2	The Edge-Edge Control Plane	74
4.4.3	The User-User Control Plane	76
4.5	Conclusion	76
5	Traffic Regulation	79
5.1	Introduction	79
5.2	The Traffic Regulation Model	81
5.2.1	Traffic Regulation Methods	81
5.2.2	Routing Functions	83
5.3	Service Models	85
5.3.1	Equal Services	85
5.3.2	Differentiated Services	86
5.4	Fairness Models	89
5.4.1	The Traffic Regulation Problem	89
5.4.2	Mono-path Fairness Characteristics	90

5.4.3	Multi-path Fairness Characteristics	92
5.5	Simulation Analysis	94
5.6	Conclusion	98
6	Conclusion	103
6.1	Thesis Summary	103
6.2	Future Work	104
Appendix		105
.1	Minimum Potential Delay	105
.2	Max-Min Fairness	106
.3	Proportional Fairness	107

List of Tables

2.1	Link utilization distribution: 50 node network	30
2.2	Back-up paths	33
2.3	100-node network: path multiplicity	33
2.4	Link utilization distribution: 50 node using FLD and KSP under moderate load (load factor 1.0)	34
2.5	Link utilization distribution: 50 node network using FLD and KSP under heavy load (load factor = 1.5)	34
3.1	LWO vs PRE-PLANNED	54
3.2	DLI vs PRE-PLANNED	55
3.3	MIXd vs MIXp	55
3.4	DCR method	56
3.5	Path utilization	57
3.6	Traffic dropped: DLI method	57
3.7	Link utilization: 100-node network	57
3.8	Path utilization: 100-node network	58
5.1	Explicit routing.	96
5.2	Explicit routing.	96
5.3	WTD using traffic halving.	97
5.4	WTD using a feedback factor 3/4.	97
5.5	WTD using different power values.	97
5.6	WTD using different queueing models.	97

List of Figures

2.1	The “trap” network	15
2.2	The “eye” network	25
2.3	Weighted traffic distribution using different power values	26
2.4	The “spoon” network	27
2.5	The “fish” network	28
2.6	Link utilization distributions	30
2.7	Path utilization distributions	30
2.8	100-node network: normalized path length distributions	31
3.1	The “link” threshold	42
3.2	The “mixed” cost function	50
3.3	Path utilization distributions	56
4.1	The “edge-edge” plane	65
4.2	The “LER” router	68
4.3	The “learning” model	73
4.4	The “edge-edge feedback” model	75
5.1	The “bottleneck” model	86
5.2	Service under different light and heavy load conditions	87
5.3	The “linear network” model	90
5.4	the “multipath fairness” model	94
5.5	the “test network” model	95

5.6	DropTail queueing model	99
5.7	RED queueing model	100
5.8	SFQ queueing model	101

Chapter 1

Introduction

1.1 Current Status of IP Networks

Communication networks have experienced a revolution during the last decade resulting from rapid developments in hardware and software which made a mixture of narrow- and wide-band applications affordable to a much larger group of users. Following this revolution, the Internet is evolving into a commercial platform requiring more than the best-effort service supported by current Internet protocols, and an expanded physical infrastructure allowing a better delivery service from IP. As a result, the Internet backbone is being upgraded into a core optical transport and many efforts are deployed to make more efficient use of IP backbone protocols by having these protocols running directly on top of optical cross-connects. Several other solutions are being investigated by the IP community to deliver better IP services in a scalable fashion. These include the move from a routing to a switching technology within the Internet backbone, the integration of enhanced protocols and QoS architectures in the network and the development of new mechanisms for resource pricing and traffic management.

1.1.1 Internet Protocols

Neither the IPv4 routing protocols currently used in the Internet nor the emerging IPv6 routing protocols provide the QoS required for real-time and bandwidth intensive applications in the modern Internet. Current generation IPv4 protocols were developed on the basis of a connection-less model where each packet's next hop and output port are determined by a computationally expensive longest-prefix-match mechanism and routing decisions are based on simple metrics such as delay or hop-count which lead to the selection of shortest-path routes. Despite its ability to scale to very large networks, this approach provides only a rudimentary QoS which does not meet the demand for scalability required for bandwidth intensive applications in modern networks. Despite its improvement over the IPv4 protocols by providing improved priority schemes, flow label fields

and extended headers, the emerging generation IPv6 protocols provide very few QoS features compared to the need for QoS in real-time applications. The design of QoS routing protocols is therefore an important issue upon which the performance of the modern Internet will depend.

1.1.2 QoS Architectures

QoS generally refers to various mechanisms aiming at providing preferential treatment to certain types of traffic with the objective of improved network performance (increased throughput, delay and jitter minimization, packet loss reduction, etc.). Several solutions from the Internet Engineering Task Force (IETF) and the research community have been proposed to support QoS in the Internet. The IETF efforts have been focused on the definition of standards for QoS mechanisms within the network while the academic, vendor and operator communities have been more involved in optimization techniques (optimal routing and optimal resource management) to be deployed in network environments supporting IETF QoS standards.

Two QoS architectures have been standardized by the IETF for QoS support in the Internet: Integrated Services (Intserv) and Differentiated Services (Diffserv) architectures (Intserv) (Diffserv). The Intserv is a QoS architecture that offers a reliable service like the circuit-switched telephone network by maintaining a per-flow state within the network. Diffserv is a stateless architecture based on the aggregated service concept supporting different service levels and allowing these different service levels to be treated differently at the ingress of the network. Various mechanisms have been proposed within the frameworks offered by the Intserv and Diffserv architectures for adding service-values to the best-effort service in order to support QoS in the Internet. These include:

- the activation of new services and resource allocation policies to meet customer needs,
- the deployment of congestion avoidance, traffic classification and prioritization mechanisms allowing an appropriate network handling to meet customer requirements and willingness to pay for services,
- network metering to support billing/accounting,
- network management and network monitoring and efficient bandwidth management to enforce bandwidth commitments to traffic sources, and
- the implementation of a flexible packet handling policy to avoid bandwidth allocations that exceed the available resources.

These mechanisms have been implemented in the Internet using various tools involving (1) queueing mechanisms such as First-In-First Out (FIFO), Priority Queueing (PQ), Classed-based Queueing (CBQ), Weighted Fair Queueing (WFQ), active queueing mechanisms such as Random Early Detection (RED) and (2) traffic shaping and policing using leaky-bucket and token-bucket mechanisms and QoS signalling using Diffserv IP precedence or RSVP mechanisms.

1.1.3 Pricing the Internet

The conversion of the Internet into a commercial platform raises the problem of resource pricing within the Internet. It is known for example that past VPN provisioning efforts deployed by the IP community failed due to the lack of an effective network infrastructure and resource pricing mechanisms that should be implemented by the Internet service providers (ISPs) to recover the costs of the investments made in the Internet infrastructure to support VPN provisioning. An emerging best-effort architecture (Sairamesh et al., 1995) (Odlyzko, 1999) (Kelly et al., 1998) (Mackie-Mason and Varian, 1995) based on pricing mechanisms is being extensively investigated by the IP community. It is based on a single class model allowing users to mark packets based on the cost of forwarding packets within the network and uses feedback mechanisms to allow users to adjust their transmission rates according to the price of the resources. Using this model, differentiation of services will be based on a market-pricing model where resources will be priced by the network based on the congestion caused by the traffic in the network, and users will implement a willingness-to-pay scheme where resources will be purchased according to the importance (priority) accorded to the traffic. This model raises the expectation that the modern Internet will still be dominated by best-effort traffic but with more control to support congestion avoidance and differentiation of services. How pricing and different congestion avoidance mechanisms will be implemented in the best-effort subnetwork is an issue which is still under investigation.

1.1.4 IP Traffic Management

Real-time traffic management has been successfully implemented in circuit-switched networks to improve the grade of service offered by a network by ensuring that the network utilization is maximized under all traffic profiles, including long-term, short-term and intermediate-term variations of the traffic. As deployed in circuit switching networks, real-time traffic management provides monitoring of network performance through a set of control mechanisms. The transition of the Internet from a non-cooperative network into a cooperative environment requires that similar control mechanisms to those deployed in real-time traffic management of circuit-switched networks be implemented in IP networks for QoS support. However, despite their successful deployment in telephone networks, real-time traffic management mechanisms have been either scarcely implemented in IP networks or scarcely find their counter-parts in IP networks. Furthermore, the traffic management mechanisms currently deployed in IP networks combining routing and resource management approaches to optimize system-wide measures of performance such as average response time, throughput, delay, etc. are inappropriate in the modern Internet: these approaches do not consider the complexity of the modern Internet characterized by the continually increasing size (number of systems and users) and heterogeneity of protocols (TCP, UDP, PPP), of applications (telnet, FTP, WEB) and resources (CPU, memory, bandwidth, servers).

1.2 Traffic Engineering

Traffic engineering is an important aspect of network management which is expected to be extensively implemented in emerging Internet protocols for real-time traffic management support. Traffic engineering allows data to be efficiently routed through the network by implementing QoS agreements between the available resources and the current and expected traffic. Multi-protocol label switching (MPLS) (Davie et al., 1998) has been proposed by the IETF to extend the IPv4 destination-based routing protocols to provide new and scalable routing capabilities including traffic engineering and traffic flow differentiation based on explicit routing. MPLS is a switching technology used by service providers and enterprises to provide connection-oriented services over connection-less network infrastructures. MPLS has been adopted by many Internet service providers (ISPs) to be used in their IP WAN network for two main reasons: the need to replace the complex and expensive overlay architecture by a more scalable architecture, and the need to provide class of service (COS), traffic engineering and virtual private networks (VPNs) at the IP layer. The overlay model was introduced as a means of upgrading the Internet backbone by allowing an IP forwarding network to be overlaid upon a switched physical network composed of ATM switches. This model requires a meshed design where the number of layer-2 circuits needed for the inter-connection of IP routers increases exponentially with the number of routers to be interconnected. This results in scalability issues which are solved by the MPLS technology by making WAN switches IP visible. By separating the routing (control functions such as route lookup) from the forwarding of IP packets, MPLS has solved the problems related to the expensive longest-prefix match used in current IP networks and the need for complicated network address translation (NAT) required to translate illegal, private or duplicate addresses. Furthermore, the ability to easily map circuit-based layer-2 QoS onto layer-3 is another feature that makes MPLS an ideal protocol for the Internet backbone.

MPLS was initially proposed by the Internet Engineering Task Force (IETF) as a topology-driven routing model which despite its relative improvement over conventional IP routing by providing higher processing speeds and differentiation of traffic flows, is constrained by the same shortest-path selection that applies in conventional IP networks. Constraint-based routing (Jamoussi, 1999) was next proposed as an improvement over the basic topology-driven MPLS to support traffic engineering by allowing packets belonging to a flow to be sent to the destination using arbitrary virtual paths (shortest and non-shortest) which have been engineered based on the flow's constraints (bandwidth guarantee, latency and jitter minimization or some other performance requirements). These virtual paths are called constraint-routed LSPs (CR-LSPs). In addition to its constraint-routing support, MPLS also provides an improved form of explicit routing compared to conventional datagram networks. Traditional datagram networks provide a form of explicit routing where the network-layer address of each node along the explicit path is attached to each packet. This results in large overheads in the packet header. In contrast, MPLS allows a less expensive form of explicit routing where label swapping is used to identify an LSP regardless of whether the LSP is established by hop-by-hop or explicit routing.

1.3 Traffic Regulation

The evolution of the Internet from a best-effort network into a commercial platform requiring resource pricing leads to a trade-off between the economic ramifications of the bandwidth sharing policy deployed in the network and the engineering efficiency achieved by the traffic regulation algorithms implemented. Network cooperation through QoS signalling is at the heart of traffic regulation mechanisms. Network cooperation allows the edge of the network to cooperate with its core to improve the performance of the system, to minimize the impact of congestion and provide end-to-end service levels and QoS policies to meet customer requirements.

QoS signalling is a form of network communication which allows an end-application to signal its bandwidth requirements and QoS requirements to the network by conveying the information in the packet header as in the IP precedence schemes used for differentiated QoS or by using a signalling protocol such as RSVP. Optimality through the use of feedback mechanisms and QoS support through the use of fair rate control schemes are the two most important aspects of traffic regulation. How different transmission rates are allocated to different paths in the best effort sub-network is therefore an important aspect that affects the performance of traffic regulation schemes and the QoS achieved by the network. Feedback mechanisms have been widely used in ATM for congestion control of ABR traffic. The ECN protocol was extensively investigated by the Internet community to support feedback mechanisms allowing applications to adjust their transmission rate to the observed network load. Feedback schemes are classified into two categories referred to as explicit and implicit feedback schemes. Implicit feedback also called bit-based feedback schemes are based on a binary indication of congestion issued by the network to allow users to adapt their transmission rate to an estimation of the network load. These schemes require minimal processing from the network: minimal participation is required from core routers and switches and minimal exchange of information between the network and users/applications. Explicit feedback schemes involve a distributed computation of rate allocations where the transmission rates are computed by the network and sent to users/applications as feedback information to adapt their transmission rate to an estimation of the network load. Though moving the processing overhead toward the network, this scheme achieves performance properties such as efficiency, fairness, controlled queueing delay and robustness. The integration of feedback schemes based on feedback mechanisms in the existing and emerging routing architectures raises an issue related to the localization of the processing entities in the routing environment and the nature of the signalling protocols implementing feedback mechanisms.

1.4 Related Work

1.4.1 Traffic Engineering

The traffic engineering problem may be addressed using either a traffic distribution approach consisting of an optimal mapping of the offered traffic to the available resources (flow routing) or a capacity engineering approach (also called capacity routing) consisting of sizing the network to optimize network resources. While traffic distribution relates to flow allocation problems, capacity engineering is more concerned with capacity allocation problems.

Traffic Distribution Approaches

Several traffic distribution proposals based either on a shortest-path or a multi-path routing scheme for adding traffic engineering capabilities to the best-effort Internet were proposed by IP researchers.

- *Shortest-path routing schemes:* Shortest-path routing schemes achieve traffic engineering by modification of link metrics to adjust the mapping of the traffic to resources. The routing model presented in (Fortz and Thorup, 2000) consisting of engineering the IP traffic by optimizing OSPF weights is a good illustration of the application of shortest-path routing in IP traffic engineering.

Despite their relative simplicity and the advantage of being piggy-backed on existing protocols, these approaches are well suited to off-line environments where traffic demands are known in advance and may lead to random traffic shifts which, if uncontrolled, may lead to performance degradations.

- *Multi-path routing schemes:* Multi-path traffic engineering approaches consist of the computation of several paths for routing the traffic offered to a source-destination pair and load-balancing these paths to increase the network performance.

The requirements for traffic engineering in MPLS networks are presented in (Awduche et al., 1998).

A first attempt to implement multi-path routing in the Internet was made in the OSPF Equal Cost Multi-path protocol (ECM) which consisted of load-balancing the traffic between several paths by halving the traffic carried by a congested path and re-routing half of this traffic over an alternate path. This scheme is based on an unbalanced traffic distribution which may lead to oscillation by continually shifting the overload from a congested path to an uncongested path.

Optimized Multi-path and MPLS Multi-path were later proposed by Villamizar (Villamizar, 1999) to correct the unbalanced distribution of traffic experienced by the ECM protocol and avoid the related problem of oscillation.

Widjaja (Widjaja, 1998) proposed MATE, a traffic engineering scheme based on periodically probing multiple paths and redistributing the traffic in order to balance loads.

Capacity Engineering Approaches

Proposals for bandwidth allocation referred to as capacity engineering (Lai, 2000) and several approaches for sharing the available bandwidth of a network between virtual networks have been presented by the Internet Engineering Task Force (IETF) (Ash, 2001).

Capacity routing approaches to traffic engineering have been extensively studied using non-linear and linear models to maximize a reward function or minimize a penalty function expressing the performance required from the network. Many heuristics for solving capacity routing problems have been proposed in the literature:

- MENTOR was presented in (Kershenbaum et al., 1991) as an heuristic algorithm solving a minimum cost topology computation problem for a mesh network applicable to the problem of obtaining starting topologies for other network design procedures.
- Flow deviation (Fratta et al., 1973) and proximal decomposition (Chifflet et al., 1994) were proposed as heuristic methods solving the loading problem by using the average packet delay as minimization objective.
- In (Wang and Wang, 1999), the capacity routing problem is formulated as an optimization problem with a capacity scale factor in the objective function.
- The re-routing heuristic algorithm proposed in (Barahoma, 1996) solves the classical network loading problem using discrete link capacity assignments.
- A cost minimization loading problem applied to IP networks which explicitly considers the restrictions imposed by the OSPF routing protocol is presented in (Benmohamed et al., 1998).
- In (Kodialam and Lakshman, 2000) Kodialam and Lakshman present an algorithm for the dynamic routing of bandwidth guaranteed tunnels in on-line environments based on a minimum interference routing concept which uses the idea that a newly routed tunnel must follow a route that does not interfere too much with a route that may be critical to satisfy a future demand.

1.4.2 Traffic Regulation

Traffic regulation of datagram traffic has been implemented following three different approaches. These include approaches deployed for ATM ABR traffic, non-pricing adaptive rate algorithms and the emerging IP pricing mechanisms implemented in the congestion control of the TCP protocol stack.

Non-pricing Algorithms

Traffic regulation is strongly related to the concept of network fairness and the adaptive bandwidth sharing algorithms used to share the network bandwidth between competitive flows. Network fairness is related to how the link bandwidth is shared between shortest and longest paths. Four fairness criteria have been presented in the literature: Maximum throughput fairness, Max-min fairness, Proportional fairness, Minimum potential delay fairness and Weighted shares fairness.

- An explicit algorithm converging in a finite number of iterations to an exact max-min fair rate allocation is presented in (Charny et al., 1995). It is based on a distributed scheme where users progressively discover their rate allocation by comparison with the advertised rate of the links on its route.
- The study presented in (Massoulié and J.Roberts, 2000) by Massoulié and Roberts is based on an alternative approach avoiding the complexity of explicit rate calculations by implementing either an end-to-end window control or rate adjustments performed by users in response to binary congestion signals to derive algorithms mimicking the four different fairness criteria.
- A more general adaptive rate algorithm including the four fairness criteria presented above was studied by Mo and Walrand in (Mo and Walrand, 2000).
- Jain and Chiu (Jain and Chiu, 1989) studied the impact of various feedback control models on the sharing of a single link between competitive flows. Their work resulted in the Additive Increase/Multiplicative Decrease (AIMD) algorithm deployed in the congestion control of the TCP protocol.

Market-pricing Algorithms

IP pricing mechanisms are subdivided into three models referred to as flat-pricing, usage-sensitive pricing and congestion-sensitive.

1. *Flat-pricing*: Pricing of Internet access is currently dominated by a flat-pricing structure based on an equal service model where a fixed monthly fee was paid by users to access the link to the network. While being questioned by several researchers, the implementation of a flat-pricing is still believed to be an acceptable choice for pricing the modern Internet (Odlyzko, 1999). This belief is based on the opinion that the current Internet is over-engineered to accommodate even peak loads and likely to remain that way.
2. *Usage-sensitive pricing*: The deployment of a usage-sensitive pricing structure is emerging where resources are priced according to the QoS required by Internet applications. This mechanism is based on the opinion that scarce resources such as bandwidth and buffer space must be priced differently according to the grade of service (GoS) received by these applications to promote a rational use of resources. Several studies addressing resource management using a usage-sensitive pricing schemes have been investigated in the literature.

- **QoS Pricing Model**

QoS pricing is presented in (Sairamesh et al., 1995) and (Sairamesh, 1997) as an approach to pricing, optimal resource allocation and QoS provisioning in high-speed packet networks. It is based on an economic model where selfish users are competing for network resources using economic agents to purchase resources in a network where network providers set prices based on demand and available supply.

- **Paris Metro Pricing (PMP) Model**

PMP is a path-differentiated service model based on a usage-sensitive pricing scheme presented by Odlyzko (Odlyzko, 1999). It is based on a “keep it as simple as it is” principle proposing the separation of the IP backbone infrastructure into several logically separated channels, each with a different price per byte allowing user applications to select for each packet which channel to send it on with the expectation that more expensive channels would attract less traffic and therefore would be less congested. PMP is a simple approach addressing only the pricing part: no assumption is made about network optimization in PMP except the need to make the network appear as simple as possible to the users.

- **Expected Capacity Contract Model**

The expected capacity contract was proposed by Clark (Clark, 1996) as a usage-sensitive pricing scheme implementing a bandwidth allocation model where users pay a price for a high probability of delivery for a given volume of traffic. In this approach user applications are charged for using the network capacity instead of being charged for actual usage: user applications whose actual usage exceeds their expected capacity during congestion periods will experience a delay in their transmission instead of having to pay more than their contracted capacity costs.

3. *Congestion-sensitive pricing*: Congestion pricing is emerging as a new pricing approach extensively investigated by the Internet community combining congestion avoidance with resource pricing to achieve network optimality. Congestion pricing is based on the implementation of an equal-service for all users under light and moderate load conditions and a differentiated service based on pricing under heavy load conditions.

- **Smart Market Pricing Model**

Smart market pricing (Mackie-Mason and Varian, 1995) proposes a packet-level pricing model which implements a willingness-to-pay scheme allowing individual packets to carry the price the sender is willing to pay to have that packet sent safely. From a set of n packets transmitted on a link which can accommodate m packets ($m \leq n$), a network implementing the smart market will process the m highest bid packets and drop the $n - m$ other packets. In a smart market scheme where the m th highest bid packets are admitted on a link, the $m - 1$ st bid is identified to be the price to be charged to packets $1, \dots, m$. A smart market is in equilibrium when the price paid for sending an additional packet referred to as the marginal cost is equal to the cutoff price imposed by the network. Despite its relative simplicity, the smart market suffers from several

implementation problems including a high transaction overhead to implement “market clearing” at potential bottlenecks in the network, a posteriori pricing unsuitable to users and the problem of packet re-ordering.

- **Proportional Fair Pricing Model**

Proportional fair pricing (Kelly et al., 1998) was proposed as a variant of congestion control mechanisms for the TCP protocol. It borrows the market-pricing concept from the smart market and the additive increase/multiplicative decrease properties from the TCP protocol. In proportional fair pricing, the network charges $p(y)$ per unit of flow transmitted at a rate of y packets per unit time. In this flow-based pricing approach, each user sending traffic in the network at a rate x is charged $xp(y)$. That user will attempt to bring the network’s charge into equilibrium with its bids by adjusting its transmission rate through an additive increase/multiplicative decrease feedback control scheme.

1.5 Thesis Outline and Main Contributions

Various frameworks have been proposed by the the research community to study QoS provisioning mechanisms to be deployed in IP networks at different time scales. The time scales include long-, medium- and short-term variations of the traffic. We propose an integrated approach combining traffic regulation and traffic engineering mechanisms in a path-oriented routing architecture layered over a Diffserv network as a solution to the problems of traffic management in IP networks. This approach proposes traffic engineering for long- and medium-term variations of the traffic and traffic regulation for short-term variations of the traffic.

The evolution of IP QoS architectures predicts an evolution of the Internet into a network including two subnetworks: a QoS guaranteed subnetwork implementing circuit-switching based QoS mechanisms using an Intserv architecture and a controlled best-effort subnetwork layered above a Diffserv network where pricing and non-pricing mechanisms will be deployed to provide the GoS required by different applications.

The design of a scalable routing architecture and the localization of different traffic management mechanisms in this routing environment is a second issue addressed in this thesis. How traffic engineering and traffic regulation are implemented in this architecture is the third issue investigated in this thesis.

1.5.1 Key Contributions

The main contributions of this thesis are threefold.

- First, we propose traffic and bandwidth allocation mechanisms to be deployed for traffic engineering multi-service logical paths (MPLS LSPs). These different techniques are integrated into a unified framework combining load-balancing and differentiated services to

support network provisioning and network reconfiguration to restore the network optimality under perturbation.

- The second contribution consists in the design of a decentralized routing architecture that controls the network at two levels:
 - long- and medium-term variations of the traffic using global control mechanisms executed in an edge-edge control plane based on an edge-core interaction to compute optimal network parameters during network provisioning (set-up and reconfiguration), and
 - short-term variations of the traffic using local control mechanisms deployed in a user-user control plane to regulate the traffic entering the network using a user-core interaction.
- The third contribution consists in the design and evaluation of traffic regulation mechanisms deployed in the two control planes using an early congestion detection model based on a threshold network loading scheme to effect a fair and effective allocation of the network bandwidth to competing traffic flows.

1.5.2 Thesis Outline

This thesis is subdivided into two parts containing the two main topics of IP traffic management addressed: Traffic engineering and traffic regulation. Traffic engineering approaches are presented in the first part which includes chapter 2 and chapter 3. The second part including chapter 4 and chapter 5 contains traffic regulation mechanisms.

Chapter 2 addresses traffic distribution aspects of traffic engineering. Chapter 3 proposes a threshold routing approach to traffic engineering. The proposed routing architecture is presented in chapter 4. Chapter 5 presents mechanisms deployed in the regulation of the traffic under long-, medium- and short-term variation of the traffic profile. The conclusions are presented in chapter 6.

Chapter 2

Traffic Engineering Model

2.1 Introduction

Traffic engineering is an important aspect of network management allowing data to be efficiently routed through the network by effecting QoS agreements between the available resources and the current and expected traffic. Multi-protocol label switching (MPLS) extends the IPv4 destination-based routing protocols to provide new and scalable routing capabilities including traffic engineering and traffic flow differentiation based on explicit routing. MPLS was initially proposed by the IETF based on a topology-driven routing model which despite its relative improvement over conventional IP routing by providing higher processing speeds and differentiation of traffic flows, is constrained by the same shortest-path selection that applies in conventional IP networks. Constraint-based routing was next proposed as an improvement over the basic topology-driven MPLS to support traffic engineering by allowing packets belonging to a flow to be sent to a destination using arbitrary paths which were selected based on the flow's constraints such as bandwidth guarantee, latency and jitter minimization or other performance requirements. These virtual paths are called constraint-routed LSPs (CR-LSPs). Constraint-based routing (CR) provides more scalable network operation by off-loading the network administrator from the task of monitoring the state of the network and executing routing and compensation mechanisms when problems arise. Constraint-routed LSPs raise three important issues: the identification of the LSPs, the distribution of the traffic among the LSPs which preserves the QoS required for the traffic being routed through the network, and the support of fault-tolerance and congestion control.

Distributing the network traffic across network links while supporting recovery from congestion and fault-tolerance is one of the main objectives of traffic engineering. This can be achieved in a multi-path environment by re-routing the traffic over alternate paths upon failure or congestion. Although the re-routing ability of a network increases with the number of paths available for re-routing, the identification of a large number of paths may be prohibitively expensive.

Supporting traffic engineering in multi-path environments is therefore a task that must be carried out carefully to avoid the performance degradation that could occur from an inefficient path computation and management mechanism.

This chapter addresses the following questions related to the optimal distribution of traffic flows to pre-computed paths in a constraint routing environment:

- how is a set of paths computed,
- how is a subset of the computed paths selected to carry the offered traffic,
- how is the offered traffic distributed among the selected paths, and
- how useful are the remaining unselected paths as backups to be used in the event of traffic overload and/or path failure

We present three path identification schemes to find path sets. A subset of these paths – the best paths in terms of the link cost metrics – is used as active paths to carry the offered traffic. The unselected paths are reserved as back-up paths to carry re-routed traffic. We describe two traffic distribution methods for load-balancing the set of active paths for carrying the traffic offered to a network and a service differentiation model based on traffic prioritization.

The remainder of this chapter is organized as follows. In section 2.2 we describe the basic flow optimization model. section 2.3 describes solution characteristics. Section 2.4 presents experimental results. The conclusions are presented in section 2.5.

2.2 A Basic Flow Optimization Model

Path identification and traffic distribution are the two most important processes involved in flow optimization. A flow optimization model may be based on either a reactive scheme where path identification and traffic distribution are computed simultaneously to achieve an optimal traffic flow allocation, or a pre-planned model where path identification and traffic distribution are performed separately.

We consider a flow optimization model based on three concepts: the separation between path identification and traffic distribution, the incremental assignment of the offered traffic to identified paths, and the separation of the set of identified paths into active paths and back-up paths. We present a pre-planned flow optimization model which consists of three steps;

Path identification. Path sets are computed for each I-E (Ingress-Egress) pair using one of three methods.

Path selection and activation. The path set is partitioned to yield a set of active paths to carry the offered traffic and a set of back-up paths for fault recovery and/or congestion avoidance. Active paths are identified either statically by selecting a fixed set of active paths or dynamically by pre-selecting a few active paths and activating new paths when the offered traffic cannot be accommodated by the available active paths.

Traffic distribution. We consider two models: a proportional traffic distribution model which shares the offered traffic among the paths according to weights expressing the loading of these paths, and a flow deviation model – a variant of the standard flow deviation algorithm – which moves the offered traffic from the highest cost paths to lower cost paths.

2.2.1 Path Identification

We present two methods for finding paths: a K shortest path (KSP) method and a topology-based shortest path (TSP) method.

A path p connecting an I-E pair (i, e) is a non-cyclic sequence of links $(\ell_1, \ell_2, \dots, \ell_h)$ connecting the ingress node i to the egress node e . For each I-E pair, the KSP method finds up to K link-disjoint shortest paths passing through the K nodes adjacent to the ingress node. Since these paths are link-disjoint, the KSP method produces different path sets depending upon the order used for the successive K path discovery. Computing the link-disjoint paths passing through the ingress neighbours selected in ascending, descending or random lexicographic order can produce three different path sets. This is illustrated by the “trap” network presented in Figure 2.1. Though there are three paths from node 1 to node 4, the KSP algorithm will find only one path if path $r_{(1,2,3,4)}$ is selected first. Otherwise, at most two paths will be found by the KSP algorithm.

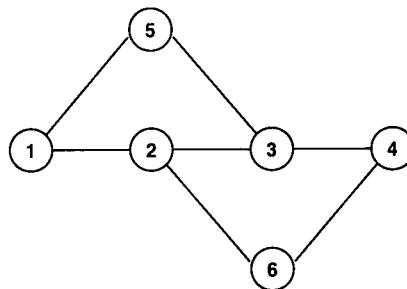


Figure 2.1: The “trap” network

The TSP method finds the K shortest paths passing through the K nodes adjacent to the ingress node. Despite its relative simplicity, this method may produce poor path sets since different paths may share links. Non-link-disjoint paths – may produce poor performance in fault recovery and present traffic bottle-necks under heavy loads.

2.2.2 Path Selection and Activation

Active Paths

Path identification may produce many paths between each I-E pair. However, practical routing environments usually use only a subset of paths to carry the offered traffic. The paths are therefore ranked according to selected metrics in order to differentiate between active paths and back-up paths. The metrics are: link capacity; hop count – this metric allows the selection of less resource consuming paths represented by shorter paths; link interference expressed by the number of paths traversing this link – this metric allows a ranking of paths which reduces the probability of producing bottleneck paths in the network; and other metrics expressing routing policies or a cost function combining two or more cost functions.

We classify the paths according to their length and effective bandwidth using the following preference functions

$$\begin{aligned} G_1(p) &= 1/L_p^\alpha \\ G_2(p) &= C_p/L_p^\alpha \end{aligned} \quad (1)$$

where C_p is the effective bandwidth of path p expressed by the minimum over its link's capacities, L_p is the length of the path expressed by the number of hops in path p and α is a positive value expressing a penalty related to the length of path p .

The preference function G_1 is better suited to traffic distribution schemes where the traffic is distributed among the least length paths. The preference function G_2 is better suited to traffic distribution schemes where the traffic is distributed among the least length paths and over-provisioned paths.

Back-up Paths

The preference functions (1) are used to partition the set of paths into two path sets corresponding to two logical networks: an active network composed of the set of nodes and links traversed by the paths that have been allocated traffic – these paths are called active paths, and a back-up network composed of nodes and links traversed by paths that have been reserved for failure recovery and congestion control.

Back-up paths may be used either as emergency paths or replacement paths when the network experiences a failure or congestion. These paths are activated when one or more active paths experiencing a link or node failure have to be replaced, and/or the number of paths carrying the traffic for a given I-E pair are not sufficient to carry the offered traffic.

Dynamic Path Activation

In a routing model based on the pre-computation of paths to carry the offered traffic and a logical separation between active paths carrying the traffic and back-up paths reserved for fault-tolerance and congestion recovery, the performance of the routing algorithm may be influenced by how the active paths are activated from the set of pre-computed paths. Two path activation models are considered: static and dynamic path activation. In static path activation, potential active paths are pre-selected before the traffic is distributed among these paths. Dynamic path activation is based on a path set augmentation scheme that starts with few paths and increases the number of paths in a path set by activating a new path from the back-up network when an existing path set cannot accommodate its offered traffic. Dynamic path activation is based on the concept of path set load expressing a load level for which the traffic offered to a source-destination pair cannot be accommodated without overloading the path set. The path set load can be expressed by three loading parameters: the path set bottleneck bandwidth, the path set threshold and the path set utilization.

- The *path set bottleneck* $C_{i,e}$ is the bandwidth of the minimum cut set expressed by:

$$C_{i,e} = \min_{\ell \in \mathcal{L}_{i,e}} (C_\ell - f_\ell) \quad (2)$$

where $\mathcal{L}_{i,e}$ is the set of links in the path set $\mathcal{A}_{i,e}$, f_ℓ is the traffic flow on link ℓ and C_ℓ is the capacity of link ℓ .

- The *path set utilization* $U_{i,e}$ is the minimum link utilization of the path set defined by:

$$U_{i,e} = \min_{\ell \in \mathcal{L}_{i,e}} \frac{f_\ell}{C_\ell} \quad (3)$$

- The *path set threshold* $T_{i,e}$ is the number of paths in a path set which are above a defined threshold load level. This number is given by:

$$T_{i,e} := |\mathcal{A}_{i,e}^*| \quad (4)$$

where $\mathcal{A}_{i,e}^* = \{p \in \mathcal{A}_{i,e} | U_p \geq T_p\}$ and $U_p = \min_{\ell \in p} f_\ell / C_\ell$ and T_p is the threshold load level associated with path p .

We observe that:

- The activation of paths based on a path set bottleneck bandwidth allows the implementation of a reactive approach where path activation is performed when the path set is congested. Path activation based on a path set utilization allows the implementation of a preventive approach where path set augmentation is triggered by a threshold which may be different from the congestion level of the path set.

- When implemented, the algorithms based on the path set utilization and path set threshold do not require prior knowledge of the traffic offered to the path set and provide a form of early detection where congestion may be detected earlier and resilience actions taken beforehand to avoid over-stressed links.

Path activation algorithms using these thresholds are presented in subsection 2.2.3.

2.2.3 Traffic Distribution

The Routing Problem

Consider a directed network of N nodes with index set \mathcal{N} and L links with index set \mathcal{L} . Let $r_{i,j}$ denote the traffic offered to the node pair (i, j) . We wish to find an optimal link flow vector \mathbf{f}_{opt} such that

$$P(\mathbf{f}_{\text{opt}}) = \min_{\mathbf{f}} \sum_{\ell \in \mathcal{L}} P_{\ell}(f_{\ell})$$

subject to the constraints

$$f_{\ell} \leq C_{\ell}$$

for all $\ell \in \mathcal{L}$ where f_{ℓ} is the average flow on link ℓ , C_{ℓ} is the capacity of link ℓ , $\mathbf{f} = (f_1, f_2, \dots, f_L)$ and $P_{\ell}(f_{\ell})$ is a link penalty function.

If the penalty function $P_{\ell}(f)$ is convex, and if the penalty function $P_{\ell}(f)$ depends only on the total flow on the link ℓ , and if the derivatives of the penalty function exist and are continuous and non-negative, and if all additional constraints that exist are included in $P_{\ell}(f)$ as penalty functions then the routing problem can be formulated as a constrained nonlinear multi-commodity flow problem.

The mathematical programming literature provides general techniques for solving multi-commodity flow problems. However, the straightforward application of these techniques to the routing problem in large networks proves to be computationally intractable. The following section presents efficient heuristic methods to solve the routing problem using a two step process consisting of path identification followed by traffic distribution.

Path Identification

The cost of a link ℓ carrying a flow f is given by

$$L_{\ell}(f) = \frac{d}{df} P_{\ell}(f).$$

The cost of a path p is given by

$$L_p = \sum_{\ell=1}^h L_{\ell}(f_{\ell}) \tag{5}$$

where h is the number of links in path p .

- **The KSP algorithm**

The KSP algorithm identifies a set of link-disjoint paths, $\mathcal{P}_{i,e}$, connecting an I-E pair. The algorithm works as follows:

for each I-E pair (i, e)

1. Set $\mathcal{P}_{i,e} := \emptyset$. For all links $\ell \in \mathcal{L}$ set the traffic flow over link ℓ , $f_\ell := 0$, and compute the cost $L_\ell(0)$ of link ℓ .
2. Compute the adjacency set \mathcal{A}_i of the ingress node i .
3. Continue to the next I-E pair if $\mathcal{A}_i = \emptyset$.
4. Find the least cost path p connecting the I-E pair (i, e) .
5. $\mathcal{P}_{i,e} := \mathcal{P}_{i,e} \cup p$.
6. Set the costs of each link in p to ∞ .
7. $\mathcal{A}_i := \mathcal{A}_i \setminus n$ where the node $n \in p$ is adjacent to i .
8. Go to step 3.

Different KSP path sets can be computed by changing step (4) to select a node $n \in \mathcal{A}_i$ in ascending, descending or random lexicographic order and then finding the least cost path p from node i to node e passing through node n .

- **The TSP algorithm**

The TSP algorithm starts with the computation of shortest paths for all I-E pairs. The subsequent steps are identical to the KSP algorithm except for step (4) which becomes

4. Select a node $n \in \mathcal{A}_i$ and find the the least cost path p from node i to node e that passes through node n .

and step (6) which is deleted.

Traffic Distribution

Two traffic distribution algorithms are considered: flow deviation (FLD) and weighted traffic distribution (WTD). We begin with a description of the standard flow deviation algorithm.

The standard flow deviation algorithm (Kereshbaum, 1993) executes in a loop computing a sequence of link flows which converges to an optimal set of link flows. At each step of the loop the link costs are computed, the shortest paths are computed, and a new flow pattern is formed by moving a portion of the existing flow from the current paths to the shortest paths. Either the flows or the link capacities may have to be adjusted so that the new links flows are feasible: a link flow is feasible if it is less than the link capacity.

Our flow deviation model differs from the standard flow deviation algorithms presented in the literature. Our FLD algorithm works in a loop. Since the paths are pre-assigned, our algorithm avoids the expense (of computational complexity $O(N^3)$) of computing the current set of shortest paths at each step of the loop. At each step of the loop a traffic increment is assigned to a least cost path so that the link flows are always feasible. A new flow pattern is formed by moving a portion of the flow either (a) from the most expensive path to the other paths, or (b) from other paths to the least expensive path as is done in the standard flow deviation algorithm.

The WTD algorithm also works in a loop. At each step, a traffic increment is distributed among the existing paths based on a fair allocation where each path receives a share of the traffic corresponding to its actual load.

We next proceed to a discussion of the three components of the traffic distribution algorithms namely incremental loading, traffic moves and the algorithms themselves.

1. *Incremental loading.* Our traffic distribution model incrementally assigns the offered traffic among the pre-computed paths. Each traffic increment is expressed in terms of a traffic multiplier. We distinguish two methods of calculating the traffic multipliers: a static scheme where a different traffic multiplier is allocated to different traffic requirements, users or applications; and a dynamic scheme where the traffic multiplier is dynamically adjusted to effect an optimal trade-off between the available network bandwidth, the application bandwidth requirements and the network performance requirements.

(a) **Static increments**

We define a static scheme where the traffic increment $a_{i,e}$ that is transmitted from an ingress node i to an egress node e is given by

$$a_{i,e} = m_{i,e} r_{i,e}$$

where $r_{i,e}$ is the traffic offered to the I-E pair (i,e) , and $0 < m_{i,e} \leq 1$ is the traffic multiplier.

(b) **Dynamic increments**

It is not known a priori what the optimal traffic multipliers are. Large multiplier values may lead to overloaded links while small values may degrade the speed of convergence of the traffic distribution algorithm by increasing the number of steps required to allocate many small increments of traffic to the selected paths. The optimal value of the traffic multiplier may be found by allowing the traffic increments to be computed dynamically. Two dynamic algorithms are considered depending on whether static path activation or dynamic path activation based on path set augmentation is used.

- i. *Static Path Set Algorithms* are based on distributing the traffic among a pre-selected number of paths using a dynamic traffic increment. Dynamic

traffic increments may be computed either by halving the traffic offered to an I-E pair or by setting the traffic increment value to the minimum slack of the least cost path of the I-E pair. This is implemented by the following algorithms:

The traffic halving algorithm is implemented as follows:

for each I-E pair (i, e) :

- A. $a_{i,e} := r_{i,e}$
- B. $compute(C_{i,e})$
- C. while $(a_{i,e} \geq C_{i,e})$
 $a_{i,e} := a_{i,e}/2$

where $C_{i,e}$ is the bottleneck bandwidth of the path set $\mathcal{P}_{i,e}$.

The minimum slack algorithm is implemented as follows:

for each I-E pair (i, e) : $a_{i,e} := \min_{\ell \in \mathcal{L}_{i,e}} (C_{\ell} - f_{\ell}) - \epsilon$

where $\mathcal{L}_{i,e}$ is the set of links in the active path set $\mathcal{A}_{i,e}$ connecting the I-E pair (i, e) and $(\epsilon \approx 0)$ is a small value used to avoid links with a zero slack.

- ii. *Path Set Augmentation Algorithms* are based on the path set load computation and the following algorithm to activate a new path $p \in \mathcal{B}_{i,e}$:

activate(i,e)

- select the least cost path $p \in \mathcal{B}_{i,e}$
- $\mathcal{A}_{i,e} := \mathcal{A}_{i,e} \cup p$

The path set bottleneck (PSB) algorithm is implemented as follows:

for each I-E pair (i, e) :

- A. $a_{i,e} := r_{i,e}$
- B. $compute(C_{i,e})$
- C. while $(a_{i,e} \geq C_{i,e})$
if $\mathcal{B}_{i,e} = \phi$ then $a_{i,e} := a_{i,e}/2$ else $activate(i, e)$

where $C_{i,e}$ is the path set bottleneck bandwidth of the path set $\mathcal{P}_{i,e}$ given by equation (2), $\mathcal{A}_{i,e}$ is the active path set and $\mathcal{B}_{i,e}$ is the back-up path set of the path set $\mathcal{P}_{i,e} = \mathcal{A}_{i,e} \cup \mathcal{B}_{i,e}$

The Path Set Threshold (PST) algorithm is implemented as follows:

for each I-E pair (i, e) :

- A. $a_{i,e} := r_{i,e}$
- B. $compute(T_{i,e})$

C. while ($T_{i,e} \geq |\mathcal{A}_{i,e}|/2$)
 if $\mathcal{B}_{i,e} = \phi$ then $a_{i,e} := a_{i,e}/2$ else *activate*(i, e)
 where $T_{i,e}$ is the path set threshold given by equation (4).

The Path Set Utilization (PSU) algorithm is implemented as follows:

for each I-E pair (i, e) :

A. $a_{i,e} := r_{i,e}$

B. *compute*($U_{i,e}$)

C. while ($U_{i,e} \leq T_{i,e}^*$)

if $\mathcal{B}_{i,e} = \phi$ then $a_{i,e} := a_{i,e}/2$ else *activate*(i, e)

where $U_{i,e}$ is the path set utilization given by equation (3) and $T_{i,e}^*$ is a given load level.

We observe that:

- In a routing environment where the traffic is offered to the least cost path, the minimum slack algorithm may guarantee the feasibility of flows at the path set level. It is not guaranteed, however, that the minimum slack algorithm will perform better in terms of the speed of convergence of the algorithm.
- The path set augmentation algorithms based on halving the traffic increment may perform better than the minimum slack algorithm in terms of the convergence of the algorithm but requires controls to guarantee the feasibility of flows at the path set level.

2. Traffic moves

The traffic offered to a path set may be distributed either based on a decrease algorithm implementing a “one-to-many” function where the traffic is moved from the highest cost path to the lower cost paths or based on an increase algorithm implementing a “many-to-one” function where the traffic is moved from the higher cost paths to a least cost path. We consider two types of traffic moves related to two distribution schemes: weighted traffic distribution (WTD) and flow deviation (FLD).

(a) Flow Deviation

- The **traffic decrease version** of the flow deviation model assigns a traffic increment to the least cost path in $\mathcal{P}_{i,e}$ and then moves an amount of flow

$$\delta_p = \alpha \frac{L_{p_0} - L_p}{L_{p_0}}$$

from the highest cost path p_0 to each of the other paths p where $p_0, p \in \mathcal{P}_{i,e}$. α is a step-size given by

$$\frac{1}{N} \leq \alpha \leq \frac{L}{N^2}.$$

- The **traffic increase version** of the flow deviation model assigns a traffic increment to the least cost path in $\mathcal{P}_{i,e}$ and then moves an amount of flow

$$\delta_p = \alpha \frac{L_p - L_{p_0}}{L_p}$$

from each of the higher cost paths p to the least cost path p_0 where $p_0, p \in \mathcal{P}_{i,e}$. α is a step-size given by

$$\frac{1}{N} \leq \alpha \leq \frac{L}{N^2}.$$

The traffic increase version is employed in the standard flow deviation algorithm.

(b) *Weighted Traffic Distribution*

The weighted traffic distribution model shares the traffic increment among the paths according to weights which express the loading of these paths. Each path $p \in \mathcal{P}_{i,e}$ is allocated a weight $w_p = 1/L_p$ inversely proportional to its length L_p as defined in equation (5). A portion of the traffic increment

$$\delta_p = \frac{w_p^\beta}{\sum_{k \in \mathcal{P}_{i,e}} w_k^\beta} a_{i,e}$$

where $\beta \geq 0$ is assigned to path p . Thus more traffic is allocated to low cost paths and less traffic to high cost paths.

3. Algorithms

- (a) **The flow deviation** algorithm works in a series of steps. Let R denote the total traffic offered to the network. In flow deviation the traffic may be moved either from the highest cost path to the the lower cost paths to effect a traffic decrease or from the higher cost paths to the least cost path to realize a traffic increase. These two algorithms are presented below.

Decrease(i,e)

- find the highest cost path $p_0 \in \mathcal{A}_{i,e}$
- **for each path** $p \in \mathcal{A}_{i,e} \setminus p_0$
 - compute the flow δ_p to be moved from path p_0 to path p
 - if $(r_{p_0} - \delta_p) > 0$:
 - i. $r_{p_0} := r_{p_0} - \delta_p$
 - ii. $r_p := r_p + \delta_p$
 - iii. for each link $\ell \in p_0 : f_\ell := f_\ell - \delta_p$
 - iv. for each link $\ell \in p : f_\ell := f_\ell + \delta_p$

Increase(i,e)

- find the least cost path $p_0 \in \mathcal{A}_{i,e}$
- **for each path** $p \in \mathcal{A}_{i,e} \setminus p_0$

- compute the flow δ_p to be moved from path p to path p_0
- if $(r_p - \delta_p) > 0$:
 - i. $r_p := r_p - \delta_p$
 - ii. $r_{p_0} := r_{p_0} + \delta_p$
 - iii. for each link $\ell \in p_0 : f_\ell := f_\ell + \delta_p$
 - iv. for each link $\ell \in p : f_\ell := f_\ell - \delta_p$

The flow deviation algorithm is presented below:

initialization:

- compute the path set $\mathcal{P} := \cup_{i,e} \mathcal{P}_{i,e}$ using the KSP or TSP algorithms: for all $p \in \mathcal{P}$ set the path flow $r_p := 0$, for all $\ell \in \mathcal{L}$ set the link flow $f_\ell := 0$.
- compute for each I-E pair (i, e) a minimal path set $\mathcal{A}_{i,e}$

iteration: while $(R > 0)$

- **for each I-E pair (i, e)**
 - compute the flow increment $a_{i,e}$
 - activate a new path if required
 - allocate $a_{i,e}$ to the least cost path $p \in \mathcal{A}_{i,e}$
 - execute Decrease(i,e) or Increase(i,e).
 - $R = R - a_{i,e}$

(b) **Weighted Traffic Distribution**

The weighted traffic distribution also works in a series of steps. Let R denote the total traffic offered to the network.

initialization:

- compute the path set $\mathcal{P} := \cup_{i,e} \mathcal{P}_{i,e}$ using the KSP or TSP algorithms: for all $p \in \mathcal{P}$ set the path flow $r_p := 0$, for all $\ell \in \mathcal{L}$ set the link flow $f_\ell := 0$.

iteration : while $(R > 0)$

- **for each I-E pair (i, e)**
 - compute the flow increment $a_{i,e}$
 - activate a new path if required
 - **for each path $p \in \mathcal{P}_{i,e}$:**
 - * compute the flow δ_p to be allocated to path p
 - * $r_p := r_p + \delta_p$
 - * for each link $\ell \in p : f_\ell := f_\ell + \delta_p$
 - * $R = R - \delta_p$

2.3 Solution Characteristics

The path identification and traffic distribution algorithms have an impact on the performance provided by the solution of the flow optimization model and the operational requirements to be satisfied. The number of active paths carrying the traffic, the re-routing of traffic under failure and the performance obtained in terms of delay, throughput are parameters which should be considered carefully to realize the best trade-off between operational requirements and performance objectives.

2.3.1 WTD Optimality

The Power Factor β

The idea behind the weighted traffic distribution algorithm (*WTD*) using the power β of the path costs is to provide a general traffic distribution factor that shares the traffic between shorter and longer paths differently according to the value of the power β . An analysis of the WTD traffic distribution using the “eye” network presented below shows that weighted traffic distribution may share the traffic differently between the shortest and longest paths depending on the value of the power β .

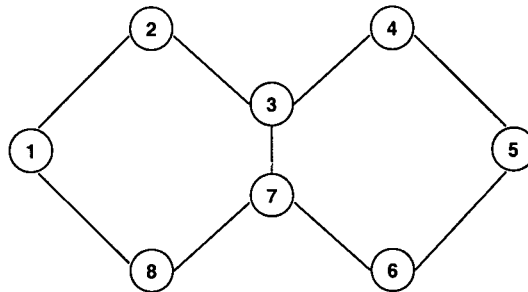


Figure 2.2: The “eye” network

Consider the “eye” network presented in Figure 2.2 where traffic is offered from node 8 to node 4 using TSP or KSP computed routes. In the situation where all links have the same slack $S = (C - f)$ and the same capacity C , using the $M/M/1$ link penalty function, the weights allocated to the shortest and longest paths respectively are:

$$w_{(8,7,3,4)} = \frac{\left(\frac{S^2}{3C}\right)^\beta}{\left(\frac{S^2}{3C}\right)^\beta + \left(\frac{S^2}{7C}\right)^\beta}$$

$$w_{(8,1,2,3,7,6,5,4)} = \frac{\left(\frac{S^2}{7C}\right)^\beta}{\left(\frac{S^2}{3C}\right)^\beta + \left(\frac{S^2}{7C}\right)^\beta}$$

If all links have capacity 1 and initially zero flow, the weights assigned to the traffic flow from

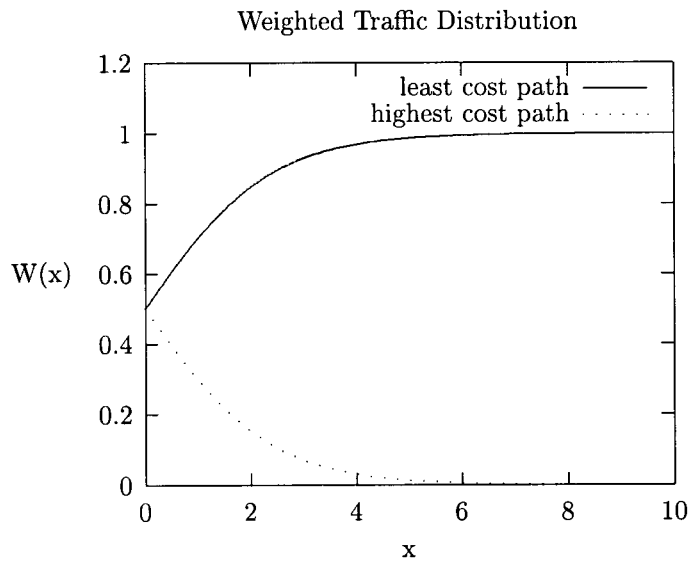


Figure 2.3: Weighted traffic distribution using different power values

source 8 to destination 4 are:

$$w_{(8,7,3,4)} = \frac{(0.33)^\beta}{(0.33)^\beta + (0.14)^\beta}$$

$$w_{(8,1,2,3,7,6,5,4)} = \frac{(0.14)^\beta}{(0.33)^\beta + (0.14)^\beta}$$

Figure 2.3 shows the effects of these weights.

- for $\beta \rightarrow \infty$, WTD implements a shortest path routing policy where longer routes are heavily penalized compared to shorter routes.
- for $\beta \rightarrow 1$, WTD implements a proportional sharing policy where longer routes are less penalized compared to shorter routes.
- for $\beta \rightarrow 0$, WTD implements an equal share policy where long and short routes receive the same flow.

Dynamic Power Values

Operational and performance requirements may restrict the number of active paths computed by the WTD algorithm either to a minimum number of paths or a maximum number of paths depending on the load conditions under consideration. Consider the “spoon” network model in Figure 2.4 where each link has capacity $C = 4$ and the traffic x offered from node 1 to node 5 may be routed over one of the two equal paths $r_{(1,2,3,5)}$ or $r_{(1,2,4,5)}$ or shared equally between the two paths.

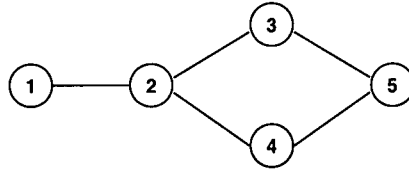


Figure 2.4: The “spoon” network

It can be observed that using the $M/M/1$ link penalty function, the total delays $D_{(1)}$ and $D_{(2)}$ obtained respectively by using a mono-path routing model sending the traffic x over one path or a multi-path routing model distributing the traffic equally among the two paths are equal for $x = 45.5$ and $D_{(1)} \geq D_{(2)}$ for $x \geq 45.5$. This result reveals that for the “spoon” network:

- multi-path routing performs better than mono-path routing only under heavy load conditions, and
- under light load conditions, a lower delay is obtained from a mono-path routing model.

The results presented above show that WTD realizes mono-path routing for high values of β ($\beta \rightarrow \infty$) and multi-path routing for lower values of β . When combined with the results that show that better performance is obtained from mono-path routing under light load and from multi-path routing under higher load, this suggests the implementation of a variant of the WTD algorithm based on dynamic power values. Traffic distribution using this dynamic model in a long-term traffic variation environment will adjust β starting with high values to realize mono-path routing under light load and decreasing β to realize multi-path routing under heavy load.

2.3.2 FLD Optimality

FLD Solutions

An evaluation of the characteristics of the solution provided by the “fish” network model presented below shows that a non-unique optimal solution may be found by the traffic distribution.

Consider the “fish” network presented in Figure 2.5 where traffic is offered from nodes 1 and 2 to node 6. The traffic requirements are $r_{(1,6)} = 0.5$ and $r_{(2,6)} = 1.5$. All links have capacity $C_i = 2$. Using the $M/M/1$ link penalty function, the optimal link flows computed by the *FLD* algorithm are

$$\begin{aligned}
 f_{(1,3)} &= 0.5 & f_{(2,3)} &= 1.5 \\
 f_{(3,4)} &= 1.0 & f_{(3,5)} &= 1.0 \\
 f_{(4,6)} &= 1.0 & f_{(5,6)} &= 1.0.
 \end{aligned}$$

Similar results are computed by the *WTD* algorithm for any value of β .

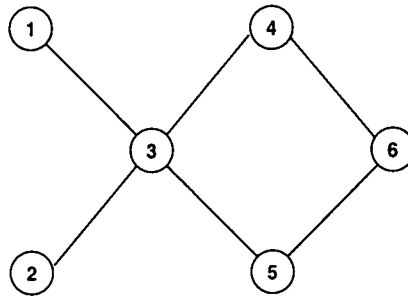


Figure 2.5: The “fish” network

The Number of Paths

It can be observed, however, that different solutions (expressed by the number of active paths identified by the algorithm and the distribution of the traffic to paths) exist to the problem of flow allocation in the “fish” network. Let r_p denote the flow on path p . We can assign any flow z where $0 \leq z \leq 0.5$ to the path (1, 3, 4, 6) whereupon the flows assigned to other paths are

$$\begin{aligned} r_{(1,3,4,6)} &= z & r_{(1,3,5,6)} &= 0.5 - z \\ r_{(2,3,4,6)} &= 1.0 - z & r_{(2,3,5,6)} &= 0.5 + z. \end{aligned}$$

It is probably an advantage for an I-E pair to have two paths rather than one path. Having four paths rather than three is probably a disadvantage. Operational requirements may prefer a particular value of z . Thus $z = 0$ and $z = 0.5$ will reduce the number of paths from four to three. The flow deviation algorithm yields $z = 0.25$ which assigns two paths from each of nodes 1 and 2 to node 6 with equal bandwidth. From the point of view of robustness under traffic forecast error, this may be the preferred solution.

Given the link flows, we need methods to compute not only a set of paths and a set of path flows consistent with the link flows, but we also need criteria to determine which set of paths and path flows are superior, and we need mechanisms to find optimal (according to those criteria) path sets and path flows.

2.3.3 Fault-tolerance

The fault-tolerance capability of a routing scheme increases with the number of paths available for re-routing the traffic and with the quality of these paths. Link-disjoint routes offer better back-up paths than link-interfering routes. It is therefore expected that KSP provides better fault-tolerance capabilities than TSP: all KSP paths connecting an I-E pair are link-disjoint whereas the TSP paths connecting an I-E pair may share links. Besides the path interference and the number of paths available for re-routing the traffic, the back-up network bandwidth and the lengths of the back-up paths are other important criteria to be considered in the evaluation of the fault-tolerance capabilities of a routing model. Thus shorter paths (expressed in terms of number

of hops) are less resource consuming than longer paths. It is therefore an advantage to distribute the traffic to link-disjoint paths only when these paths are not excessively long.

Flow allocation mechanisms deployed in multi-path traffic engineering schemes appeal to a tradeoff between fault-tolerance and efficiency. It was claimed above that carrying the traffic over link-disjoint paths provides increased fault-tolerance. It can be shown, however, that fault-tolerant solutions are not always the most efficient solutions in terms of resource usage, network optimization and fairness. Indeed, using the “eye” network model, KSP path identification may generate two sets of paths from node 8 to node 4: S_1 containing paths $r_{(8,7,3,4)}$ and $r_{(8,1,2,3,7,6,5,4)}$ and S_2 containing paths $r_{(8,7,3,4)}$ and $r_{(8,1,2,3,4)}$. It can be observed that the path set S_1 allocates the traffic to a long path $r_{(8,1,2,3,7,6,5,4)}$ which may be more resource consuming than using paths identified by a TSP method. It is an advantage for the traffic offered to an I-E pair to be allocated to link-disjoint paths rather than link-interfering paths for fault-tolerance. However uncontrolled fault-tolerant implementations may result in reduced throughput and increased delays. This shows the merit of selecting paths based on the preference functions G_1 and G_2 expressed by equation (2.1) to effect the cutoff of resource consuming paths.

2.4 Performance Analysis

We conducted several experiments to analyze the impact of the different path identification and traffic assignment methods on MPLS network performance in a constraint-routing environment where the cost of the network is

$$T = \frac{1}{R} \sum_{\ell \in \mathcal{L}} P_{\ell}(f_{\ell})$$

and the link cost is

$$P_{\ell}(f_{\ell}) = \frac{M f_{\ell}}{C_{\ell} - f_{\ell}} \quad (6)$$

where R is the total packet arrival rate from external sources, M is the average packet length and $M/(C_{\ell} - f_{\ell})$ is the average delay experienced by a packet traversing a link ℓ of capacity C_{ℓ} where the link is modelled as a $M/M/1$ queue.

The relevant network performance parameters are the link utilization distribution, the traffic dispersion in terms of the number of paths which have been allocated traffic, the fault-tolerance expressed by the number and quality of paths available for re-routing the traffic under failure and the speed of convergence of the algorithms expressed by the time required to compute paths and distribute the offered traffic among these paths.

The networks used in our experiments consist of 50- and 100-node test networks each carrying one traffic class. The 50-node network has 202 uni-directional links which are capacitated with 38,519,241 units of bandwidth. A total of 6,581,372 units of flow are offered to the 2,450 I-E pairs. The 100 node network has 244 uni-directional links which are capacitated with 6,515,881 units

link utilization	FLD		WTD	
	KSP	TSP	KSP	TSP
0.0-0.1	1	1	1	1
0.1-0.2	2	0	2	0
0.2-0.3	11	8	12	10
0.3-0.4	25	30	23	32
0.4-0.5	50	49	50	48
0.5-0.6	66	66	67	71
0.6-0.7	42	41	42	35
0.7-0.8	5	6	5	4
number of active paths	3,082	3,229	3,626	3,954
network delay	226	229	227	216
average utilization	0.5	0.5	0.5	0.5
standard deviation	0.12	0.13	0.12	0.13
speed of convergence (seconds)				
path identification	2.7	0.3	0.6	2.7
traffic distribution	0.5	0.5	0.6	8.5

Table 2.1: Link utilization distribution: 50 node network

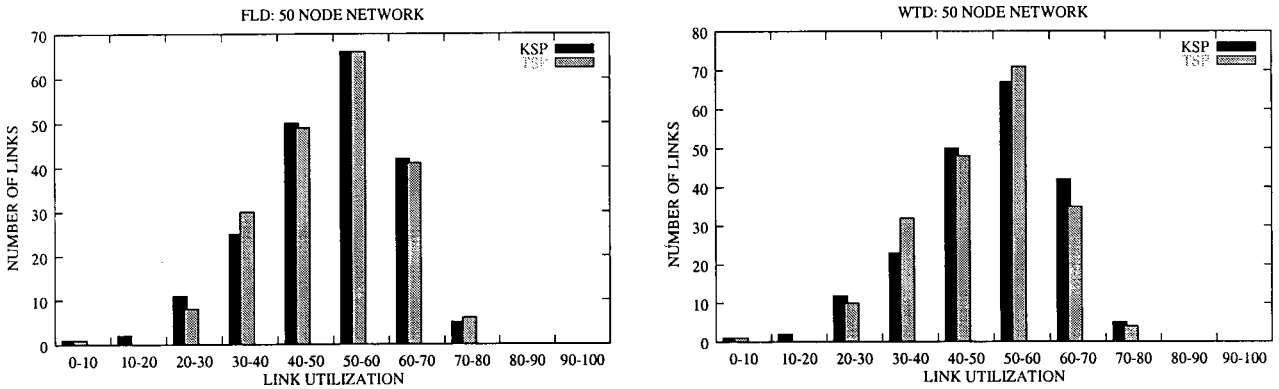


Figure 2.6: Link utilization distributions

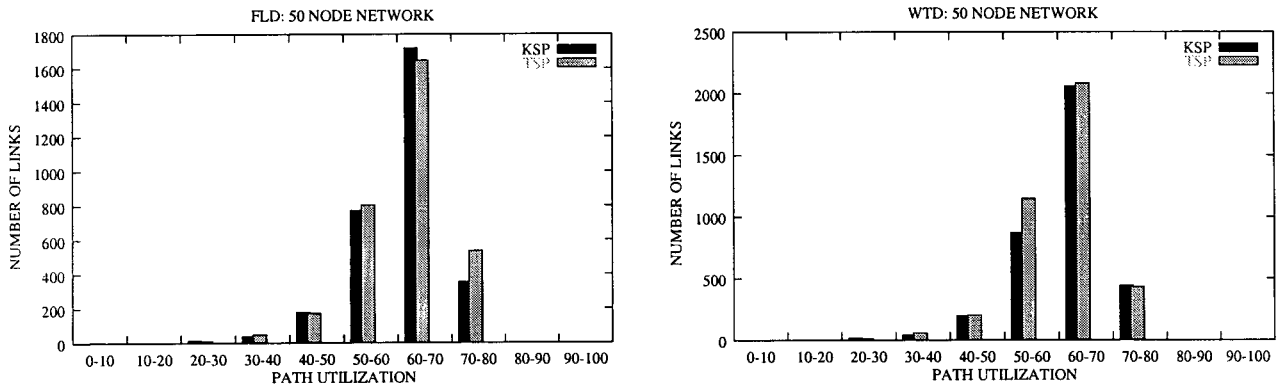


Figure 2.7: Path utilization distributions

of bandwidth. A total of 250,003 units of flow are offered to the 9,900 I-Epairs. A description of the networks with their link capacities and offered traffics can be found at the URL <http://www.cs.sun.ac.za/projects/COE/models.zip>.

Table 2.1 presents the link utilization distributions in the 50-node network where the KSP and TSP path finding methods are used and the traffic is distributed by the FLD and WTD traffic distribution models. The path utilization expresses the maximum over its link utilization. In both cases the KSP and TSP distributions agree closely. Figure 2.7 presents the path utilization distribution corresponding to Table 2.1.

Table 2.1 shows that KSP computes up to 20% fewer active paths than TSP – however, Table 2.2 shows that KSP computes many more back-up paths than TSP; KSP path computation takes substantially longer than TSP (TSP requires that the shortest paths be computed once whereas KSP requires many recalculations of the shortest paths); WTD uses up to 30% more paths than FLD; and WTD traffic distribution takes substantially longer than KSP. All computations were performed on a 700MHz Pentium III processor.

Figure 2.6 shows the link utilization distributions when the routes are computed using the KSP and TSP methods and the traffic is distributed using the FLD and WTD methods. The KSP path set produces slightly more under- and over-loaded links. Figure 2.6 shows that the KSP and TSP methods yield similar link utilization distributions for the 50-node network.

Figure 2.7 shows that the FLD method computes fewer over-utilized paths compared to FLD.

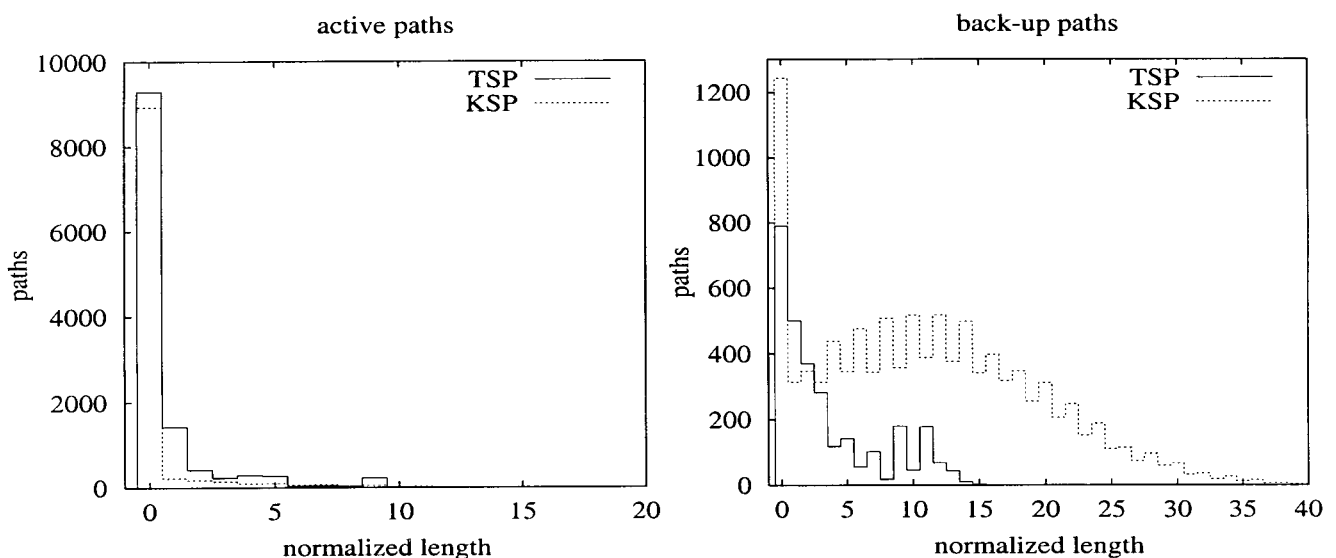


Figure 2.8: 100-node network: normalized path length distributions

Tables 2.2 and 2.3 and Figure 2.8 were presented in a study of the quality of routes computed by the flow deviation method (Bagula and Krzesinski, 2001) using the $M/M/1$ link penalty function in the 50- and 100-node network model. The length of a path denotes the hop-count

of a path. The normalized length of a path is the length (hop-count) of that path minus the length (hop-count) of the shortest path (minimal hop-count) connecting the I-E pair of that path. Figure 2.8 shows the normalized lengths of the active and the back-up paths in the 100-node network as found by the KSP and TSP methods. Nearly all the active paths have normalized length 0: the offered traffic is distributed to the shortest paths connecting each I-E pair.

The quality of a back-up route may be expressed by its normalized length and its available bandwidth indicating the potential to carry traffic. The fault-tolerance capability of a routing scheme increases with the number of back-up paths available for re-routing and with the quality of these paths. Link-disjoint routes offer better back-up paths than non-link-disjoint routes. KSP therefore provides better fault-tolerance capabilities than TSP: all the KSP paths connecting an I-E pair are link-disjoint whereas the TSP paths connecting an I-E pair may share links. Figure 2.8 shows that the KSP algorithm found many more back-up paths than the TSP algorithm, although many of the KSP back-up paths have a large normalized length and are therefore unlikely to function well as back-up paths.

This situation is examined in more detail in Table 2.2 which presents the number of back-up paths reserved by the KSP and TSP algorithms for the 50- and 100-node networks. With reference to the 100-node network, KSP finds many more back-up paths than TSP. Only 8% of the TSP back-ups are link-disjoint whereas all of the KSP paths are link-disjoint. KSP provides back-up for 99% of the I-E pairs and for 99% of the traffic. However, if we require that the back-up paths have a logical length of at most 5, then KSP provides back-ups for 32% of the I-E pairs which carry 45% of the traffic. Under these restrictions, TSP provides back up for only 12% of the I-E pairs which carry 7% of the traffic. Only 10% of the TSP back-up paths are pairwise link-disjoint. KSP also provides a richer set of back-up paths for the 50-node network. In this case, restricting the back-up paths to a logical length of 5 has no effect since all paths in the 50-node network have a logical length of less than or equal to 5.

Table 2.3 presents information concerning the traffic distribution in the 100-node network. The KSP method finds an optimal path set containing 9,919 active paths. The average normalized path length is 0.40 and the average path bandwidth is 25.2. The TSP method finds an optimal path set containing 12,696 active paths. The average normalized path length is 0.75 and the average path bandwidth is 19.7. The active path sets have several attractive features. The paths overwhelmingly coincide with the shortest paths connecting the I-E pairs. Most I-E pairs are connected by one or two paths, and nearly all of the flow is assigned to the shortest paths. Some 96% of the bandwidth is assigned to the shortest and the shortest-but-one paths.

An I-E pair is said to have an n -path connection or a path multiplicity of n if the pair is connected by n paths. Table 2.3 shows the bandwidth assigned to n -path connections where $n = 1, 2, \dots, 6$. For example the second row of Table 2.3 shows that 1,808 I-E pairs are connected by two paths: the 3,616 paths carry 47,148 units of flow which is 18.8% of the total flow carried by the network. Each of these paths carries on average 13.0 units of flow. Each two-path connection carries on

	50-nodes		100-nodes	
	TSP	KSP	TSP	KSP
active paths	3,135	2,560	12,696	9,919
back-up paths: no length restriction				
back-up paths	2,100	6,093	2,494	10,441
% link-disjoint back-up paths	38.4	100.0	8.8	100.0
% I-E pairs with back-up paths	52.7	99.9	22.7	99.9
% flow with back-up paths	52.3	99.9	12.4	99.7
back-up paths: logical length ≤ 5				
back-up paths	2,100	6,012	1,261	3,211
% link-disjoint back-up paths	38.4	100.0	9.6	100.0
% I-E pairs with back-up paths	52.7	99.9	12.2	31.5
% flow with back-up paths	52.3	99.9	7.3	44.8

Table 2.2: Back-up paths

average 26.0 units of flow. The OPT and KSP methods yields an average path multiplicity of 1.28 and 1.00 respectively.

We conducted experiments to evaluate the impact of the traffic increment on the network performance under moderate and heavy load by multiplying the offered traffic by a load factor K . A load factor $K = 1$ is referred to as moderate network load while a load factor $K = 1.5$ represents heavy network load. The results are presented in Table 2.4 and Table 2.5. The results presented in table 2.4 show that under moderate load, the traffic increment has little impact on the network delay achieved. However, a lower traffic dispersion (fewer active paths found) and a relatively high speed of convergence of the traffic distribution algorithm are achieved under larger traffic increments.

The network delay is more affected by the size of the traffic increment under heavier loads. This is shown in Table 2.5 where relatively larger delays are computed under larger traffic increments. Table 2.5 presents the same variation of the number of active paths inversely to the traffic increment as in Table 2.4.

paths	I-E pairs	routes	flow	%	flow/route	flow/I-E
TSP method: network delay 94.5						
1	7,630	7,630	192,925	77.1	25.2	25.2
2	1,808	3,616	47,148	18.8	13.0	26.0
3	404	1,212	7,915	3.1	6.5	19.5
4	54	216	1,813	0.7	8.4	33.5
5	2	10	44	0.0	4.4	22.2
6	2	12	81	0.0	6.8	40.9
KSP method: network delay 97.0						
1	9,881	9,881	247,274	98.9	25.0	25.03
2	19	38	2,727	1.0	71.7	143.57

Table 2.3: 100-node network: path multiplicity

link utilization	traffic increment				
	0.5	0.25	0.125	0.10	0.05
0.0-0.1	1	1	1	1	1
0.1-0.2	3	2	2	2	2
0.2-0.3	7	10	11	12	11
0.3-0.4	32	25	24	23	25
0.4-0.5	46	49	50	50	50
0.5-0.6	64	67	65	66	66
0.6-0.7	45	43	44	43	42
0.7-0.8	4	5	5	5	5
number of active paths	2,782	2,946	3,028	3,049	3,082
network delay	228	227	226	226	226

Table 2.4: Link utilization distribution: 50 node using FLD and KSP under moderate load (load factor 1.0)

link utilization	traffic increment				
	0.25	0.125	0.05	0.025	0.02
0.0-0.1	0	0	0	0	0
0.1-0.2	0	0	0	0	0
0.2-0.3	0	0	0	0	0
0.3-0.4	4	5	5	5	5
0.4-0.5	8	9	8	7	8
0.5-0.6	14	16	19	21	20
0.6-0.7	25	25	21	20	19
0.7-0.8	42	36	37	37	38
0.8-0.9	51	60	64	65	66
0.9-1.0	58	51	48	47	46
number of active paths	3,358	3,259	3,585	3,597	3,610
network delay	1,438	1,319	1,271	1,262	1,260

Table 2.5: Link utilization distribution: 50 node network using FLD and KSP under heavy load (load factor = 1.5)

2.5 Conclusion

This chapter evaluates the use of a pre-planned flow optimization model to find label switched paths in an MPLS network and to optimally distribute the offered traffic among these LSPs.

We present two path finding methods. The K shortest path method KSP computes a link-disjoint path set for each I-E pair. The TSP method computes a path set for each I-E pair that is not necessarily link-disjoint. We present two traffic distribution algorithms: flow deviation (FLD) and weighted traffic distribution (WTD).

A limited evaluation using 50- and 100-node test networks of the path finding and traffic distribution algorithms allows us to conclude that the KSP and FLD algorithms efficiently compute near-optimal active paths sets and traffic distributions while identifying a useful set of back-up paths that can be used in the event of failure among the active paths.

Chapter 3

Threshold Routing

3.1 Introduction

Distributing the traffic across links to optimize the network utilization and maximize the potential for traffic growth is one of the main objectives of traffic engineering which can be achieved by implementing either a shortest-path routing or a multi-path routing approach. Different issues related to each of these two approaches are addressed in this chapter:

- Multi-path routing approaches based on distributing the traffic among multiple paths may result in an internal fragmentation of the network resulting from the presence of several paths with small residual bandwidth that cannot accommodate a new request though the aggregate bandwidth is sufficient to satisfy the new request. Furthermore, network topologies do not always provide multiple paths to carry the traffic between each source-destination pair. It is therefore suggested that mono-path schemes implementing a congestion avoidance approach be deployed as alternatives to multi-path routing approaches to traffic engineering.
- The shortest-path approach to traffic engineering provides poor performance compared to multi-path routing approaches and leads to unexpected behaviour such as uncontrolled traffic shifts. Furthermore, this approach is based on an opportunistic routing model which does not support policy-routing. Hybrid approaches combining the best of shortest-path routing and multi-path routing approaches need to be defined.
- Threshold routing aims at moving the traffic away from over-stressed links to avoid the internal path fragmentation of multi-path routing or to effect a better link utilization in mono-path routing approaches. This is achieved by a traffic shaping model which can be successfully implemented only in sufficiently-provisioned networks. Methods to compute the additional bandwidth required to effect an efficient traffic shaping have to be designed.

The remainder of this chapter is organized as follows. In section 3.2 we describe the threshold routing approach to traffic engineering. Section 3.3 presents methods used in performance measurement. In section 3.4 we describe a threshold-differentiated services model. Section 3.5 contains experimental results. The conclusions are presented in section 3.6.

3.2 The Threshold Routing Approach

The main objective of threshold routing is to keep the link loads within their capacities while effecting the best distribution of the traffic across links. This can be achieved by the implementation of a balanced distribution of the traffic across the network using congestion avoidance mechanisms which move the traffic away from congested paths to less utilized parts of the network.

We consider two threshold routing models: (1) Link weight optimization (LWO) and (2) Differentiated Routing Regime (DRR).

- *Link weight optimization (LWO)* maps link utilization levels into link weights which are used in routing updates.
- *Differentiated Routing Regime (DRR)* protects the network from link/path over-load by either routing the traffic over alternate paths or sizing the link capacity to meet predefined link utilization levels expressed in terms of link thresholds.

3.2.1 Link Weight Optimization(LWO)

Open Shortest Path First (OSPF) is one of the most commonly used protocols for intra-domain routing in the Internet. It is based on a model where the routing is determined by a static approach using one weight (a measure of the cost) for each link. The LWO approach is based on a dynamic approach mapping link utilization levels into link weights used in the routing process.

The LWO problem

Consider a directed network of N nodes with index set \mathcal{N} and L links with index set \mathcal{L} , a threshold vector $\mathbf{t} = (t^1, t^2, \dots, t^K)$ representing a discrete set of values expressing the load levels reached by the network and a link cost vector $\mathbf{c} = (L^1, L^2, \dots, L^K)$ associated with the threshold vector. Let C_i denote the capacity of link i and $r_{i,j}$ denote the traffic offered to node pair (i, j) . Let $A_{i,j}$ denote the set of paths which carry the traffic offered to the node pair (i, j) and f_r denote the flow carried by path r .

We wish to find an optimal link flow vector \mathbf{f}_{opt} such that

$$P(\mathbf{f}_{\text{opt}}) = \min_{\mathbf{f}} \sum_{\ell \in \mathcal{L}} P_{\ell}(f_{\ell})$$

subject to the constraints

$$\sum_{r \in A_{i,j}} f_r = r_{i,j} \quad (7)$$

$$\frac{d}{df} P_{\ell}(f_{\ell}) = L^i, \text{ for } t^i \leq f_{\ell}/C_{\ell} < t^{i+1} \quad (8)$$

and

$$f_{\ell} \leq C_{\ell} \quad (9)$$

We observe that:

- In the LWO formulation, the derivative of the link penalty with respect to the traffic flow $\frac{d}{df} P_{\ell}(f)$ is defined as a piece-wise constant function. The corresponding link penalty is thus a piece-wise linear function of the traffic flow which if increasing and convex will lead to a unique solution solved in polynomial time.
- Equation (7) expresses that the flows $r_{i,j}$ are carried.
- Equation (8) states that the link costs used in the routing updates are derived from the link thresholds and equation (9) expresses the feasibility of the link flows.

LWO updates

In the LWO routing model, each link ℓ is assigned a link cost derived from its load level and the path cost is given by the sum over its link cost expressed as:

$$L_p = \sum_{\ell \in p} \frac{d}{df} P_{\ell}(f) \quad (10)$$

LWO Algorithms

Two algorithms may be used for routing the traffic using the LWO model: a shortest path routing algorithm and a multi-path routing algorithm using a variant of the WTD or FLD algorithms presented in chapter 2.

The shortest-path and WTD versions of the LWO model start by computing the set of paths for each I-E pair using the TSP or KSP algorithm. Further steps are implemented as follows:

Shortest-path version

While $R > 0$:

- for each I-E pair (i, e) :
 1. compute the path costs using equations (8) and (10)
 2. find the least cost path $p \in \mathcal{A}_{i,e}$,
 3. compute the traffic increment $a_{i,e}$,
 4. assign $a_{i,e}$ to p
 5. $R := R - a_{i,e}$

WTD version

While $R > 0$:

- for each I-E pair (i, e) :
 1. compute the path costs using equations (8) and (10)
 2. compute the traffic increment $a_{i,e}$,
 3. for each path $p \in \mathcal{A}_{i,e}$
 - compute the traffic δ_p to be allocated to path p
 - adjust the link costs
 4. $R := R - a_{i,e}$

where R is the total traffic offered to the network, $a_{i,e}$ is the traffic increment computed in chapter 2 and $\mathcal{A}_{i,e}$ is the active path set associated with the I-E pair (i, e) .

3.2.2 Differentiated Routing Regime (DRR)

DRR implements a congestion-avoidance approach which protects the network from link/path over-load by marking network links/paths with thresholds expressing the load level reached by these links/paths and implementing different routing regimes (different routing objectives and cost functions) under different load levels.

Two routing regimes corresponding to two loading conditions are considered:

1. a light load regime corresponding to light load conditions where no penalty related to the path over-load is applied to the network links, and
2. a threshold regime corresponding to heavy load conditions where a penalty related to the path over-load is applied to the network links.

Four DRR routing approaches are considered:

- *Path-differentiated routing* implements an end to end model which routes the offered traffic based on path thresholds expressing the maximum load levels to be supported by pre-computed paths.
- *Dynamic Link Isolation* routes the offered traffic based on link thresholds expressing the maximum load levels to be supported by links. Dynamic Link Isolation is a hop by hop model which allocates an infinite cost to threshold-marked links to isolate these links from receiving further traffic.
- *Dynamic Traffic redistribution* adapts the link cost to the actual network load in order to move the traffic away from over-stressed links by allocating an additional cost to threshold-marked links.
- *Dynamic Capacity Resizing* increases the capacity of links which have reached the threshold level to maintain the link load level below a defined threshold.

Path-differentiated Routing

1. The Path-differentiated Routing Problem

Consider a directed network of N nodes with index set \mathcal{N} and L links with index set \mathcal{L} , a set of path load levels (service classes) \mathcal{S} and a threshold vector $\mathbf{t} = (t^1, t^2, \dots, t^S)$ associated with the service classes. Let \mathcal{A}_ℓ denote the set of paths which traverse a link ℓ and $t(r) \in \mathbf{t}$ denote the threshold for traffic on path r . Let C_i denote the capacity of link i . Let $r_{i,j}$ denote the traffic offered to node pair (i, j) . Let $A_{i,j}$ denote the set of paths which carry the traffic offered to the node pair (i, j) and f_r denote the flow carried by path r .

We wish to find an optimal link flow vector \mathbf{f}_{opt} such that

$$P(\mathbf{f}_{\text{opt}}) = \min_{\mathbf{f}} \sum_{r \in \mathcal{R}} \sum_{\ell \in r} P_\ell(f_\ell^r)$$

subject to the constraints

$$\sum_{r \in A_{i,j}} f_r = r_{i,j} \quad (11)$$

$$\frac{f_\ell}{C_\ell} \leq \min_{r \in \mathcal{A}_\ell} \{t(r)\} \quad (12)$$

and

$$\sum_{r \in \mathcal{A}_\ell} f_\ell^r \leq C_\ell \quad (13)$$

for all $\ell \in \mathcal{L}$ where f_ℓ^r is the flow on path r flowing over link ℓ , the link flow $f_\ell = \sum_{r \in \mathcal{A}_\ell} f_\ell^r$, the link vector $\mathbf{f} = (f_\ell)_{\ell \in \mathcal{L}}$ and $P_\ell(f_\ell^r)$ is a link penalty function where $f^r = \{f_\ell^r\}_{\ell \in r}$.

We observe that:

- The objective function presented above does not involve any differentiation between traffic classes.
- Equation (11) expresses that the flows $r_{i,j}$ are carried.
- Equation (12) states that the links cannot be loaded above the threshold of the path carrying the most QoS-demanding traffic (traffic that requires the lowest path threshold). The threshold of the most QoS-demanding path serves as cutoff for the traffic flowing over a link.
- Equation (13) expresses the feasibility of the link flows.

2. Routing Updates

The path-differentiated routing model may be mapped into a link model by mapping the path thresholds into link thresholds as proposed below. In that case, routing updates follow the DLI model.

3. Mapping path into link thresholds

It can be observed from equation (12) that the threshold $t(r)$ of a path r constitutes a cut-off for the traffic flowing through the links traversed by that path. Link thresholds may thus be derived from path thresholds by assigning to a link a threshold corresponding to the minimum over the thresholds of paths traversing that link. This is illustrated in Figure (3.1) where the threshold of link $\ell = (3, 4)$ is given by the minimum over the thresholds of the two paths r_1 and r_2 traversing the link:

$$t(3, 4) = \min\{t(r_1), t(r_2)\} \quad (14)$$

where \mathbf{t} is the set of threshold values, $t(r_1) \in \mathbf{t}$ and $t(r_2) \in \mathbf{t}$ are the thresholds of paths $r_1 = r_{(1,3,4,5)}$ and $r_2 = r_{2,3,4,6}$ respectively.

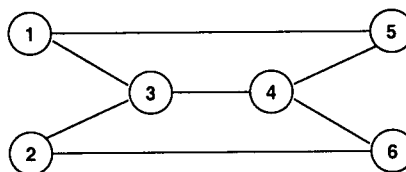


Figure 3.1: The “link” threshold

The computation of link thresholds from path thresholds consists in the mapping of path into link thresholds defined by the following equation:

$$t(\ell) = \min_{r \in \mathcal{A}_\ell} \{t(r)\} \quad (15)$$

for all links $\ell \in \mathcal{L}$.

Dynamic Link Isolation (DLI)

The Dynamic Link Isolation (DLI) model employs thresholds which express the maximum utilization supported by a link and implements a traffic distribution which isolates threshold-marked links from receiving further traffic.

The link thresholds used in the DLI model may either be derived from path thresholds used by the path-differentiated routing model presented above or defined by the network operator to express policy-routing constraints which are used to protect specific network paths or links from over-load.

1. The DLI Problem

Consider a directed network of N nodes with index set \mathcal{N} and L links with index set \mathcal{L} , a set of link load levels (classes) \mathcal{K} and a threshold vector $\mathbf{t} = (t^1, t^2, \dots, t^K)$ associated with the link load levels. Let \mathcal{A}_ℓ denote the set of paths which traverse a link ℓ and $t(\ell) \in \mathbf{t}$ denote the threshold for traffic on link ℓ . Let C_i denote the capacity of link i . Let $r_{i,j}$ denote the traffic offered to node pair (i, j) . Let $A_{i,j}$ denote the set of paths which carry the traffic offered to the node pair (i, j) and f_r denote the flow carried by path r .

We wish to find an optimal link flow vector \mathbf{f}_{opt} such that

$$P(\mathbf{f}_{\text{opt}}) = \min_{\mathbf{f}} \sum_{\ell \in \mathcal{L}} P_\ell(f_\ell)$$

subject to the constraints

$$\sum_{r \in A_{i,j}} f_r = r_{i,j} \quad (16)$$

$$\frac{f_\ell}{C_\ell} \leq t(\ell) \quad (17)$$

for all $\ell \in \mathcal{L}$ where f_ℓ is the total traffic carried by link ℓ , $\mathbf{f} = (f_\ell)_{\ell \in \mathcal{L}}$, f_ℓ^r is the flow of path r carried by link ℓ and $P_\ell(f_\ell)$ is a link penalty function where $f_\ell = \sum_{r \in \mathcal{A}_\ell} f_\ell^r$.

We observe that:

- The objective function presented above does not involve any differentiation between traffic classes.
- Equation (16) states that the flows $r_{i,j}$ are carried.
- Equation (17) states that the links cannot be loaded above their maximum load level.

2. DLI Routing updates

We consider a routing model where the link cost metrics on links are given by the derivative of the link penalty function with respect to the flows expressed by $L_\ell(f) = \frac{d}{df} P_\ell(f)$ below threshold level and an infinite cost above threshold load.

The cost of link ℓ is:

$$L_{\ell}(f_{\ell}) = \begin{cases} \frac{dP(f_{\ell})}{df_{\ell}} & : 0 \leq f_{\ell}/C_{\ell} < t(\ell) \\ \infty & : f_{\ell}/C_{\ell} \geq t(\ell) \end{cases}$$

3. The Link isolation method

The DLI problem may be expressed as the problem of implementing a policy-routing model aimed at protecting some parts of the network designated by the network operator from link over-load.

The DLI model achieves policy-routing by shaping the traffic offered to the network to meet constraints expressing the maximum link utilization required to support the bandwidth protection of these links.

In the DLI model, the link thresholds expressing the maximum load levels to be supported by network links are used as constraints in the traffic distribution process to trigger the isolation of a link from receiving further traffic when that link is threshold-marked.

The DLI problem may be solved by computing paths using the KSP or TSP algorithms and implementing a link isolation model which isolates a link that has reached its threshold from receiving further traffic by assigning an infinite cost to this link. A variant of the FLD or WTD algorithms using the link cost expression $L_{\ell}(x)$ given above instead of the link cost $\frac{d}{df}P_{\ell}(f_{\ell})$ is used to implement the link isolation method.

The link isolation method involves three steps: (1) computing paths using the KSP or TSP algorithms, (2) mapping policy-routing constraints into link thresholds, and (3) traffic distribution.

(a) *Path computation*

The path computation may be performed using either the TSP or the KSP algorithms.

(b) *Link threshold marking*

Each link involved in the policy-routing is threshold marked to implement a differentiated cost model on that link: a normal cost below threshold and a infinite cost over threshold.

(c) *Traffic distribution*

The same traffic distribution algorithms used for the FLD and WTD algorithms may be used to distribute the traffic in the link isolation approach. Only routing updates are performed based on the link cost $L_{\ell}^i(x)$ for each traffic x flowing through link ℓ .

Dynamic Traffic Redistribution(DTR)

The DTR routing method consists in the determination of an optimal set of link and path flows which minimize a given penalty function while effecting different routing regimes under different load regions by monitoring the link load to avoid over-stressed links.

1. The DTR Problem

Consider the network presented in section 3.2.2 (see path-differentiated routing problem) above. Let $t(\ell)$ denote the threshold for traffic on link ℓ .

We wish to find an optimal link flow vector \mathbf{f}_{opt} such that

$$P(\mathbf{f}_{\text{opt}}) = \min_{\mathbf{f}} \sum_{r \in \mathcal{R}} \sum_{\ell \in r} P_{\ell}(f_{\ell}^r)$$

subject to the constraints

$$\sum_{r \in \mathcal{A}_{\ell}} f_{\ell}^r \leq C_{\ell} \quad (18)$$

$$\frac{f_{\ell}}{C_{\ell}} \leq t(\ell) \quad (19)$$

$$t(\ell) = \min_{r \in \mathcal{A}_{\ell}} \{t(r)\} \quad (20)$$

for all $\ell \in \mathcal{L}$ where $f_{\ell} = \sum_{r \in \mathcal{A}_{\ell}} f_{\ell}^r$, f_{ℓ}^r is the flow of path r on link ℓ and $\mathbf{f} = (f_{\ell})_{\ell \in \mathcal{L}}$.

We observe that:

- Equation (18) expresses the feasibility of flows.
- Equation (19) states that the links cannot be loaded above their maximum load level.
- The routing constraints (20) expresses the relation between the link and path thresholds. Constraint (19) may be ignored in a policy-routing model where the link thresholds are pre-defined by the network operator.

2. DTR updates

In the DTR model, routing updates are performed through the deployment of two types of link costs: a cost defined by $\frac{d}{df} P_{\ell}(f_{\ell})$ applied before the threshold load is reached and a cost $L_{\ell}^{\phi}(f_{\ell})$ deployed after the threshold load is reached.

The link cost associated with the DTR model is:

$$L_{\ell}(f_{\ell}) = \frac{d}{df} P_{\ell}(f_{\ell}) + \alpha_{\ell}(f_{\ell}) \phi_{\ell}(f_{\ell}) \quad (21)$$

$$\alpha_{\ell}(f_{\ell}) = \begin{cases} 0 & : 0 \leq f_{\ell}/C_{\ell} < t(\ell) \\ 1 & : \text{if } f_{\ell}/C_{\ell} \geq t(\ell) \end{cases} \quad (22)$$

We observe that:

- The link cost function 21 expresses the differentiated routing regime by implementing two different routing costs : $\frac{d}{df} P_{\ell}(f_{\ell})$ for $\alpha_{\ell}(f_{\ell}) = 0$ and $L_{\ell}^{\phi}(f_{\ell}) = \frac{d}{df} P_{\ell}(f_{\ell}) + \phi_{\ell}(f_{\ell})$ when $\alpha_{\ell}(f_{\ell}) = 1$.

- Equation (22) expresses the differentiated routing regime by defining two different link load levels: a light and moderate link load level corresponding to a link utilization under threshold and a link utilization above the threshold level $t(\ell)$.

3. The threshold cost $L_\ell^\phi(f)$

The choice of the link cost deployed after the threshold load level is an important aspect of the DTR approach. We consider a threshold cost including the actual link cost $\frac{d}{df}P_\ell(f_\ell)$ and an additional cost $\phi_\ell(f_\ell)$ expressing a penalty related to the loading of the link above its threshold traffic $f_\ell^* = C_\ell t_\ell$ defined as the traffic flow corresponding to the threshold load level on link ℓ .

- The additional cost $\phi_\ell(f_\ell)$ included in the threshold cost $L_\ell^\phi(f_\ell)$ may be defined by the following equation:

$$\phi_\ell(f_\ell) = \left(\frac{f_\ell - t(\ell)}{t(\ell)} \right)^\beta \quad (23)$$

where β is a power value expressing the penalty related to the loading of a link above its threshold or similarly a reward for moving the link load below the threshold level.

- The threshold routing objective is to minimize the difference between the actual load level and the threshold load level to move the link load into a light load region where the load is below the threshold level. As expressed by equation (23), the additional cost $\phi_\ell(f)$ is a relative distance expressing this difference. The threshold cost is therefore a cost which optimizes the routing objective by deploying an additional cost which is proportional to the difference between the actual link load and its threshold level.

Dynamic Capacity Redistribution (DCR)

The DCR routing problem consists of finding an optimal set of link flows and link capacities required to minimize a given penalty function while meeting routing constraints expressed in terms of link thresholds.

1. DCR Problem

Consider the network presented in section 3.2.2 (see DTR problem) above.

We wish to find an optimal link flow vector \mathbf{f}_{opt} and an optimal capacity vector \mathbf{C}_{opt} such that

$$P(\mathbf{f}_{\text{opt}}, \mathbf{C}_{\text{opt}}) = \min_{\mathbf{f}, \mathbf{c}} \sum_{r \in \mathcal{R}} \sum_{\ell \in \mathcal{R}} P_\ell(f_\ell^r, C_\ell(f_\ell^r))$$

subject to the constraints

$$\sum_{r \in \mathcal{A}_\ell} f_\ell^r \leq C_\ell \quad (24)$$

$$t(\ell) = \min_{r \in \mathcal{A}_\ell} \{t(r)\} \quad (25)$$

for all $\ell \in \mathcal{L}$ where f_ℓ^r is the flow of path r on link ℓ , $\mathbf{f} = (f_\ell)_{\ell \in \mathcal{L}}$, and $C_\ell(f_\ell)$ is the capacity allocated to link ℓ when it is carrying the traffic f_ℓ .

We observe that:

- The objective function expresses a routing and capacity allocation model to find the optimal link flows and capacities which meet the routing constraints.
- Equation (24) expresses the feasibility of flows.
- Equation (25) expresses the the relation between path and link thresholds.

2. The capacity scaling factor ϕ_ℓ

We consider a bandwidth over-provisioning approach which resizes the capacity of a link based on the expression:

$$C_\ell(f_\ell) = (1 + \alpha_\ell(f_\ell)\phi_\ell(f_\ell))C_\ell \quad (26)$$

$$\alpha_\ell(f_\ell) = \begin{cases} 0 & : \quad 0 \leq f_\ell/C_\ell < t(\ell) \\ 1 & : \quad \text{if } f_\ell/C_\ell \geq t(\ell) \end{cases} \quad (27)$$

We observe that:

- The function $\phi_\ell(x)$ expresses the additional capacity required to guarantee the threshold utilization on link ℓ .
- The equation (26) expresses the differentiated routing model by assigning to the link capacity a constant value C_ℓ below threshold and a dynamic value $(1 + \phi_\ell)C_\ell$ above the threshold level.
- The capacity scaling factor ϕ_ℓ specifies how far the link capacity is from the capacity needed to maintain the link utilization below its defined threshold. This factor may be expressed as a relative distance $\phi(f_\ell)$ from the ideal link capacity expressed by

$$\phi(f_\ell) = \left(\frac{\frac{f_\ell}{C_\ell} - t_\ell}{t(\ell)} \right)^\beta \quad (28)$$

- The expression of the capacity scaling factor given by equation (28) is derived from the routing objective optimization consisting in the minimization of the difference between the actual link load and the threshold load level to move the link load below the threshold level. In this case the power value β also expresses a reward for moving the link load below the threshold level.

DRR Routing Algorithms

The DRR routing model uses the same algorithms as the basic flow optimization model presented in chapter 2 to compute a set of active and back-up paths and distribute the traffic among the computed paths; except the link costs are computed differently.

3.3 Performance Measurement

3.3.1 Path Characteristics

The performance of the routing process depends on an estimation of the load on available paths which is measured by the congestion of these paths. Several congestion measures may be considered. These include the link slack, the packet loss probability, the link delay derivatives, the link utilization and congestion functions combining several congestion characteristics.

- *The link slack* is defined by the difference between the capacity of a link and the traffic carried by that link and is a congestion measure that represents the bottleneck bandwidth on paths. Using this measure, the distribution of the traffic among paths may be performed by maximizing the minimum link slack on paths.
- *The derivative of the link delay* with respect to the flow is a congestion measure representing a marginal cost that may be used to quantify the sensitivity of the path to perturbation in the path load. Using this measure, the distribution of the traffic among paths is performed by minimizing the sum over the links traversed by this path of the link derivative delay.
- *The link utilization* is another congestion measure that can be used to measure the bottleneck bandwidth on a path. Traffic distribution schemes using this measure are implemented by minimizing the maximum over the link utilizations of the paths.
- *Link thresholds:* Routing in IP networks was widely driven by traffic allocation schemes where bandwidth is allocated to traffic requirements based on static metrics such as the link propagation delay or the path length expressed in number of hops. Emerging traffic engineering schemes may use threshold routing metrics allowing the link cost to be derived from the actual link load as proposed in (Fortz and Thorup, 2000) and the different routing approaches presented above. In these dynamic schemes, different costs are assigned to links for different link utilization levels.

3.3.2 Link Cost Optimization (LCO)

Modern communication networks involving QoS routing requirements lead to more complex routing models where path computation is based on the use of multiple routing metrics such as bandwidth, delay and loss probability. This differs from traditional best effort data networks which are primarily concerned with connectivity, and route the traffic based on simple metrics such as hop-count and propagation delay using shortest-path algorithms for path computation.

Multiple metrics can model QoS routing more accurately but finding a path subject to multiple constraints is inherently difficult. The alternative to QoS routing using multiple routing metrics

consists of link cost optimization models which either combine several routing metrics in a function used in routing updates or shape the form of single routing metrics to achieve the desired network performance.

Link cost metrics

Shaped single cost metrics

Shaping the form of a single cost metric may result in improved performance. A study on competitive routing of virtual circuits in ATM networks (Plotkin, 1995) has reported that assigning each link a cost that is exponential in its current utilization results in optimal blocking probability. We consider three routing models where exponential link costs are used for routing updates. These include:

1. The exponential of the link utilization:

$$L_\ell = \left(\frac{f_\ell}{C_\ell}\right)^\alpha \quad (29)$$

2. The exponential of the derivative of the link penalty:

$$L_\ell(f) = \left(\frac{d}{df}P_\ell(f)\right)^\alpha \quad (30)$$

3. The exponential of the available bandwidth on a link:

$$L_\ell = (C_\ell - f_\ell)^\alpha \quad (31)$$

where $P_\ell(f)$ is the link penalty function, α is a factor expressing either the relative penalty assigned to heavily loaded link compared to lightly loaded link or the relative penalty assigned to highest cost links compared to least cost paths, f_ℓ is the traffic flowing through link ℓ and C_ℓ is the capacity of link ℓ .

It can be observed that values of $\alpha = 0$ lead to load and cost independent routing while large values of α favour the discrimination between least cost/lightly loaded paths and highest cost/heavily loaded paths.

Mixed Link Metrics

Combining different metrics into a unique function which is used for routing updates is another solution to QoS routing. Consider the following mixed link metrics:

$$L_\ell = \left(\frac{I_\ell}{C_\ell - f_\ell}\right)^\alpha \quad (32)$$

$$L_\ell = \left(\frac{d_\ell}{C_\ell - f_\ell}\right)^\alpha \quad (33)$$

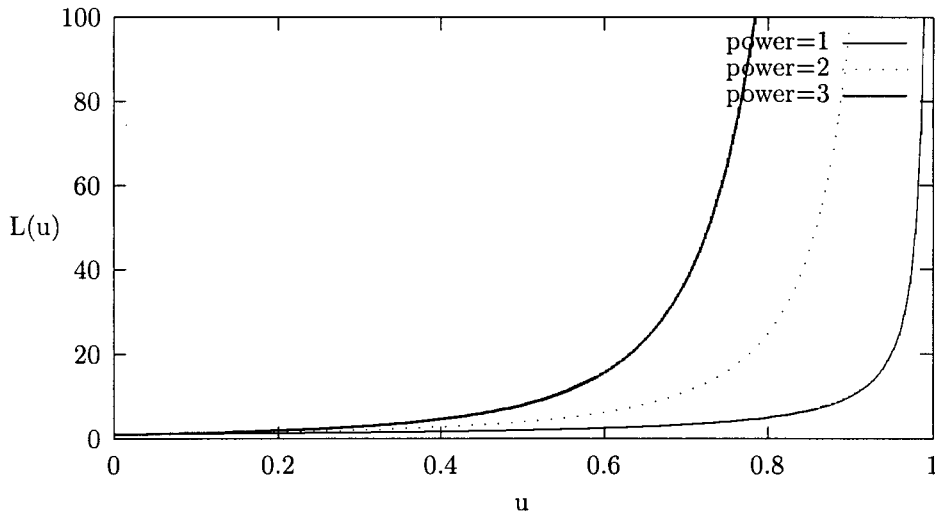


Figure 3.2: The "mixed" cost function

$$L_\ell = k_\ell + \left(\frac{t_\ell}{t_\ell - u_\ell}\right)^\alpha \quad (34)$$

where I_ℓ is the interference of link ℓ expressing the number of flows sharing that link, $u_\ell = f_\ell/C_\ell$ is the link utilization, d_ℓ is the link propagation delay and $k_\ell = d_\ell - 1$.

We observe that the link cost achieved by equation (34) equals the propagation delay d_ℓ under light loads ($u_\ell \ll t_\ell$) and a high value approximating the infinity under higher loads ($u_\ell \approx t_\ell$). Figure 3.2 shows the variation of the mixed link cost 34 with the link utilization for a threshold value ($t_\ell = 100$) and various values of the power ($\alpha = 1$, $\alpha = 2$ and $\alpha = 3$).

The relative merits of this routing metric is evaluated in the section on performance evaluation.

Path cost metrics

Two types of path cost metrics may be derived from link costs. These include the sum of the link costs and the maximum of the link costs.

The *sum* of the link costs is expressed as:

$$L_p = \sum_{\ell=1}^h L_\ell \quad (35)$$

The *maximum* of the link costs is expressed as:

$$L_p = \max_{\ell \in \{1, \dots, h\}} L_\ell \quad (36)$$

where h is the length of the path in hops and L_ℓ is the link cost.

It can be observed that:

- Summing the link costs presents the advantage of having the loads in the rest of the network being minimized independently of the presence of a bottleneck heavily loaded link.
- Routing based on the maximum link costs provides throughput guarantees for tunnelled traffic over dedicated virtual paths to meet service level agreements.

3.4 Differentiated Services Models

Two differentiated services models illustrating the two main approaches for tunnelling the IP traffic over a network infrastructure are considered: path-differentiated service model and path-multiplexed service model. The path-differentiated model illustrates a class-based tunnelling approach multiplexing several customer traffics over the same single class tunnel while the path-multiplexed model mimics a customer-based tunnelling approach multiplexing several traffic classes over a single tunnel belonging to a customer.

3.4.1 The Path-differentiated Model

Path-differentiated services have been addressed in the Paris Metro Pricing (PMP) approach as a simple mechanism supporting differentiated services over an IP network infrastructure by implementing a differential pricing model among computed logical channels with the expectation of routing less traffic over expensive channels which will therefore be less congested.

We propose a similar approach implementing a non-pricing scheme where the network infrastructure is separated into several logical channels, each with a different maximum load level expressed by a threshold to support a form of path prioritization where the traffic is re-routed over non-threshold marked paths when a path has reached its threshold load. By moving the traffic away from threshold-marked paths, the threshold routing model keeps network paths within their assigned load levels expressing the QoS provided to the traffic flowing over these paths. The idea behind path-differentiated services through threshold routing is to map QoS requirements into path load levels expressing the QoS provided to the traffic flowing along these paths. The traffic carried by low threshold-marked paths (lowest load levels) should thus experience a lower delay than the traffic carried by higher threshold-marked paths (higher load levels).

The differentiated services problem may be expressed as the problem of provisioning K VPNs $\{\mathcal{A}^k\}$ over a network infrastructure where each VPN \mathcal{A}^k has its own QoS requirements expressed in terms of maximum load level (threshold) t^k that its paths may support. This problem may be solved by finding for each VPN \mathcal{A}^k at least one path $r_k \in \mathcal{A}^k$ for each I-E pair and implementing a traffic distribution algorithm consisting in the isolation of a path that has reached its threshold

from receiving further traffic and the re-routing of the remaining traffic allocated to this path over an alternate path if an alternate path exists. The path isolation scheme may be implemented using a variant of the FLD or WTD algorithms presented above where in the initialization step at least K paths are computed by the KSP or TSP algorithms for each I-E pair and the link cost used in routing updates is given by the expression of $L_\ell(x)$ given above instead of the link cost (??).

The path isolation method involves three steps: (1) path computation, (2) mapping VPN QoS requirements into path thresholds, and (3) VPN bandwidth distribution.

1. Path computation

The VPN provisioning model is implemented by computing for each source-destination pair a number of paths equal to the number of VPNs using the KSP or TSP algorithm.

2. Path QoS marking

The mapping of the VPN QoS requirements into a path threshold is performed to assign to each of the K computed paths for each source-destination a threshold corresponding to its VPN QoS requirements: a path $p \in \mathcal{A}^k$ is assigned a threshold t^k corresponding to the maximum load level of paths belonging to VPN \mathcal{A}^k .

The performance of the VPN provisioning scheme may depend on the mapping of the QoS requirements to the computed paths. This mapping is implemented by ranking pre-computed paths based on some defined fixed cost metrics such as the minimum bandwidth available on paths, the interference of paths or other criteria and assigning thresholds to paths based on one of the two marking methods: (1) a best-to-highest marking model which marks paths proportionally according to their quality by assigning to the the best paths the highest thresholds, or (2) a best-to-lowest marking model which marks paths in the inverse order of their quality by assigning the lowest thresholds to the best paths.

3. VPN bandwidth distribution

The traffic distribution model is based on a variant of the FLD or WTD traffic distribution model allowing the traffic to be distributed as in the FLD or WTD algorithms with the exception that the the link costs are computed using the expression $L_\ell(x)$ defined above which requires the computation of a minimum link threshold based on the path thresholds. This link threshold may pre-computed based on the path threshold as proposed in the following chapter or defined by fixed values allocated by the network operator based for example on policy-routing requirements.

The path isolation method consists of a routing procedure including the following successive steps:

1. compute for each I-E pair (i, e) its path set $\mathcal{P}_{i,e}$ using the TSP or KSP algorithm presented in chapter 2

2. compute for each link $\ell \in \mathcal{L}$ its threshold using equation (12)
3. distribute the traffic among the existent paths using either the WTD_p or the FLD_p algorithms presented below.

3.4.2 The Path-multiplexed Model

Path-differentiated services raise the problem of link/path migration resulting from the transfer of links/paths from the highest toward the lowest threshold-marked VPNs. Path-multiplexing is an alternative approach which solves the problem of link/path migration by sharing the link bandwidth between several VPNs according to the following equation:

$$\sum_{r \in \mathcal{A}^k \cap \mathcal{A}_\ell} C_\ell^r \leq C_\ell \quad (37)$$

where $C_\ell^r = \frac{f_\ell^r}{t^k}$ is the bandwidth allocated to the path r belonging to VPN k on link ℓ .

The path-multiplexed problem may be formulated by the same model as the path-differentiated model except the cut-off constraint (12) which is replaced by the equation (37) expressing the link-sharing concept. The link-sharing is implemented by priority queueing mechanisms such as Class-Based Queueing (CBQ) allowing paths traversing a link to be allocated bandwidth on that link according to its threshold using, for example, a proportional allocation model which allocates bandwidth to threshold-marked paths inversely to their threshold levels: higher threshold-marked paths receive small bandwidth portions.

3.5 Performance Analysis

We conducted a set of experiments using the 50-node model to compare the LWO method using step-size cost values with the basic pre-planned flow optimization model using the “DEL” and “MIXp” cost functions. The “DEL” link cost function uses the derivative of the link delay with respect to flow plus the propagation delay as cost function. The “MIXp” cost function was presented in (Burns et al., 2001) as an objective function deployed in a non-linear optimization model achieving OSPF routing under light load and slack maximization under heavier loads. The LWO link cost function considered approximates the mixed cost function (34) where the link threshold $t(\ell) = 1$, the propagation delay $d_\ell = 1$ and the power $\beta = 3$. The results presented in Table 3.1 show that for the link cost considered, LWO computes more links in the load region 0.6 – 0.7 and many more active paths compared to the pre-planned flow optimization of chapter 2. However, pre-planned flow models achieve very high utilized links in the range of 0.9 – 1.0 which can degrade the network performance by setting bottlenecks which can experience very high link delays. In contrast, LWO does not compute very high loaded links.

A set of experiments was conducted to evaluate the relative performance gains obtained from the DLI routing model compared to the basic flow deviation method. The results of these experiments

link utilization	LWO		PRE-PLANNED	
	cost	number of links	DEL	MIXp
0.0-0.1	1	0	0	0
0.1-0.2	1.37	2	2	0
0.2-0.3	1.95	9	13	6
0.3-0.4	2.91	20	23	24
0.4-0.5	4.62	43	56	59
0.5-0.6	8.00	60	63	60
0.6-0.7	15.62	66	33	46
0.7-0.8	35.03	2	7	4
0.8-0.9	125		3	1
0.9-1.0	1,000		1	2
number of active paths		3,710	2,788	2,799
average utilization		0.53	0.51	0.52
standard deviation		0.11	0.13	0.11

Table 3.1: LWO vs PRE-PLANNED

are presented in Table 3.2. Table 3.2 compares the link distribution computed by the basic pre-planned flow optimization model using FLD to the threshold routing approach based on the DLI method using the “DEL”, “MIXd” and “MIXp” link cost functions in the 50-node network. The “MIXd” link cost function uses the proposed mixed link cost function (34). This table shows that though the DLI model using the “MIXd” cost function finds approximately 10% more active paths than the other models, this model achieves the lowest load levels: all the link utilizations are below 70%. However, the “MIXd” cost function computes more active paths compared to the other models. This results from the activation of new paths for re-routing the traffic when the current active paths reach their threshold level.

The experiments also show that the DLI models using the “DEL” and “MIXp” cost functions compute the same number of active paths than pre-planned models and do not lead any link in the load region above 90% utilization. The DLI methods achieve approximately the same values of the standard deviation as pre-planned models. The threshold models thus realize a distribution of the traffic which is as consistent with the utilization distribution achieved as are pre-planned models.

We compared the “MIXd” to the “MIXp” under various load profiles by multiplying the traffic offered to the 50 node network by a load factor K . The results presented in Table 3.3 show that routing approaches using the “MIXd” cost function achieve lower utilization levels compared to the “MIXp” cost model under all the traffic profiles. These results also show that though computing approximately 10% more active paths under light loads ($k < 1.5$), the “MIXd” approach finds approximately the same number of paths than the “MIXp” under heavy load ($k = 1.5$).

Despite their relative benefits compared to non-threshold routing methods, threshold routing methods can drop traffic at the sources to satisfy constraints on bottleneck links when implemented with low threshold level requirements. This situation is analyzed in Table 3.6 which presents the

amount of traffic dropped under various threshold level requirements by the DLI method. This table shows that the 50-node network implementing the DLI algorithm based on the “MIXp” method drops traffic on the sources when the threshold level required on each link is below 78%. Such situations give rise to a tradeoff between the adoption of a bandwidth over-provisioning strategy and the packet dropping strategy or the readjustment to high-threshold levels when network over-provisioning is not possible. Table 3.6 also shows that the “MIXd” model performs better than the “MIXp” model under high threshold requirements. The “MIXd” does not drop traffic but reacts to high threshold requirements by activating more paths.

link utilization	DLI			PRE-PLANNED	
	DEL	MIXd	MIXp	DEL	MIXp
0.0-0.1	0	0	0	0	0
0.1-0.2	2	0	0	2	0
0.2-0.3	13	5	6	13	6
0.3-0.4	23	19	24	23	24
0.4-0.5	54	53	59	56	59
0.5-0.6	62	76	60	63	60
0.6-0.7	36	49	46	33	46
0.7-0.8	7		4	7	4
0.8-0.9	5		3	3	1
0.9-1.0				1	2
number of active paths	2,816	3,184	2,801	2,788	2,799
average utilization	0.51	0.52	0.52	0.51	0.52
standard deviation	0.13	0.09	0.11	0.13	0.11

Table 3.2: DLI vs PRE-PLANNED

link utilization	1.0		1.25		1.50	
	MIXd	MIXp	MIXd	MIXp	MIXd	MIXp
0.0-0.1	0	0	0	0	0	0
0.1-0.2	0	0	0	0	0	0
0.2-0.3	5	6	0	1	0	0
0.3-0.4	19	24	5	5	2	3
0.4-0.5	53	59	13	21	6	7
0.5-0.6	76	60	29	35	15	19
0.6-0.7	49	46	44	49	23	33
0.7-0.8		4	76	36	36	33
0.8-0.9		3	35	46	62	35
0.9-1.0				9	58	72
number of active paths	3,184	2,801	3,323	2,969	3,623	3,614
average utilization	0.52	0.52	0.69	0.68	0.79	0.78
standard deviation	0.09	0.11	0.12	0.14	0.13	0.15

Table 3.3: MIXd vs MIXp

We conducted experiments to evaluate the additional capacity required to realize different threshold levels on each link of the 50-node model. The results are presented in Table 3.4. Similar

experiments can be conducted to evaluate the additional link capacity in a network where different threshold values are assigned to different links or paths. Table 3.4 presents the link utilization distribution for different threshold levels and the additional bandwidth required by the DCR algorithm to meet these thresholds in the 50-node model. These results show that the additional bandwidth required to achieve threshold levels decreases linearly with the increase of the link utilization, and DCR distributes the traffic similarly to the available paths under different thresholds.

The performance of a routing approach deployed in path-oriented environment depends on the link utilization levels achieved as well as the path utilization levels. The path utilization is defined by the maximum over its link utilizations. High path utilization levels can reduce the potential for traffic growth. Table 3.5 compares the path utilization distribution achieved by the DLI routing model using the “MIXd” with the pre-planned model in the 50-node model using the “MIXp” cost function. This table shows that the DLI routing model computes more low-utilized paths and achieves low link utilization levels compared to the pre-planned flow optimization model. These results are confirmed in Figure 3.3 which shows that the pre-planned path load levels reach utilization levels above 70% while the DLI load levels are below 70%.

link utilization	link threshold				
	0.67	0.69	0.71	0.73	0.75
0.0-0.1	1	1	1	1	1
0.1-0.2	2	2	2	2	2
0.2-0.3	13	12	12	12	12
0.3-0.4	22	23	23	23	23
0.4-0.5	50	52	50	50	50
0.5-0.6	67	65	67	67	67
0.6-0.7	43	43	42	42	42
0.7-0.8	4	4	5	5	5
additional bandwidth	71,417	37,939	18,756	6,099	0

Table 3.4: DCR method

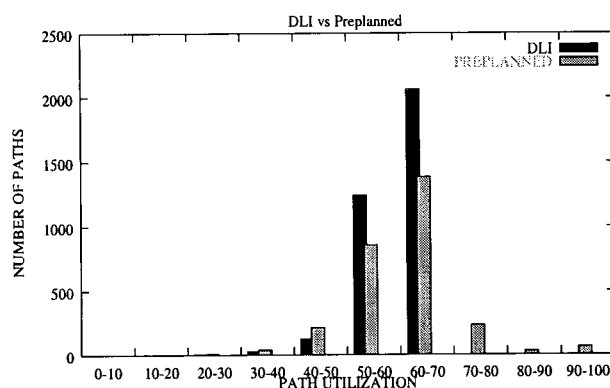


Figure 3.3: Path utilization distributions

The results presented in Tables 3.7 and 3.8 using the 100-node network confirm that the DLI

link utilization	DLI		PRE-PLANNED	
	path	link	path	link
0.0-0.1	0	0	0	0
0.1-0.2	0	0	0	0
0.2-0.3	5	5	6	6
0.3-0.4	25	19	38	24
0.4-0.5	125	53	214	59
0.5-0.6	1,242	76	855	60
0.6-0.7	2,061	49	1,383	46
0.7-0.8			232	4
0.8-0.9			29	1
0.9-1.0			61	2
number of active paths	3,184		2,799	
average utilization	0.52		0.52	
standard deviation	0.09		0.11	

Table 3.5: Path utilization

link utilization	0.66		0.70		0.74		0.78	
	MIXd	MIXp	MIXd	MIXp	MIXd	MIXp	MIXd	MIXp
traffic drop	0	28,197	0	3,112	0	261	0	131
number of active paths	3,464	3,185	3,342	2949	3,279	2,851	3,232	2,817
average utilization	0.53	0.52	0.52	0.52	0.52	0.52	0.52	0.52
standard deviation	0.09	0.11	0.09	0.11	0.09	0.11	0.09	0.11

Table 3.6: Traffic dropped: DLI method

link utilization	DLI		DTR	PRE-PLANNED	
	MIXp	MIXd		MIXp	MIXd
0.0-0.1	22	22	22	23	22
0.1-0.2	142	136	136	141	141
0.2-0.3	26	32	32	28	29
0.3-0.4	14	14	14	11	11
0.4-0.5	8	9	9	11	10
0.5-0.6	14	15	15	9	12
0.6-0.7	2	0	0	5	4
0.7-0.8	4	9	8	2	3
0.8-0.9	10	7	8	4	6
0.9-1.0	2			10	6
number of active paths	11,736	12,007	12,011	12,036	12,100

Table 3.7: Link utilization: 100-node network

path utilization	DLI		DTR	PRE-PLANNED	
	MIXp	MIXd		MIXp	MIXd
0.0-0.1	46	46	46	50	46
0.1-0.2	2,681	2,358	2,357	2,677	2,731
0.2-0.3	882	1,365	1,364	1,008	1,043
0.3-0.4	375	443	443	220	335
0.4-0.5	533	525	524	618	573
0.5-0.6	1,630	1,960	1,959	1,292	1,645
0.6-0.7	248	0	0	516	426
0.7-0.8	460	1,950	1,947	292	1,590
0.8-0.9	4,877	3,360	3,371	1,606	200
0.9-1.0	4			3,757	3,511

Table 3.8: Path utilization: 100-node network

and DTR threshold methods achieve low utilized links and paths compared to the pre-planned method. These results also reveal that the DLI and DTR methods achieve the same utilization distribution.

3.6 Conclusion

This chapter evaluates the use of a threshold routing approach to optimally distribute the traffic offered to an MPLS network among pre-computed LSPs. Thresholds expressing the maximum load assigned to the links are used both as barriers preventing the network from becoming over-loaded, and as triggers to initiate different routing regimes. We present two threshold routing methods using two mappings of the link utilization into cost metrics in order to implement different routing updates in an MPLS environment. The LWO method maps link utilizations into predefined fixed link cost metrics while the DRR method maps link utilization into dynamic link cost metrics which are used as barriers to avoid link over-load. DLI is deployed as a congestion avoidance approach to avoid network over-load by isolating threshold-marked links from receiving further traffic. By increasing the cost of threshold-marked links, DTR reroutes part of the traffic of these links over alternate paths. DCR computes the bandwidth required to over-provision the network in an environment where the network topology presents bottleneck links which need to be over-capacitated to accommodate the offered traffic.

A limited evaluation using an off-line routing model shows that threshold routing achieves low utilization rates compared to the basic pre-planned flow optimization approach presented in chapter 2.

Several routing metrics proposed in this chapter can produce poor performance compared to the mixed cost metric considered. However, operational requirements such as the type of Link State Advertisement (LSA) protocol and the type of Management Information Base (MIB) used in routing updates may prefer the use of a slightly less efficient method compared to a more efficient

method. Furthermore, methods performing poorly in an off-line environment may sometimes perform better in on-line environments.

Two differentiated service models using multiple paths per route for QoS provisioning are presented: path-differentiated and path-multiplexed services. Path-differentiated services were presented as a model for implementing service discrimination in an MPLS environment where LSPs are computed using the KSP method. Despite its simplicity, this model is better-suited to over-provisioned network environments or to routing environments implementing network over-provisioning when link constraints are not met or in routing environments where only a few priority tunnels are constructed over a network infrastructure dominated by low priority tunnels. Path-multiplexed services were presented as an alternative approach solving the restrictions related to the path-differentiated services model. The design of detailed algorithms implementing these two service models and their performance evaluation has been reserved for future research work.

Chapter 4

Proposed Routing Architecture

4.1 Introduction

Devising effective routing architectures to provide an appropriate quality of service (QoS) to a mixture of narrow- and wide-band traffic is a challenging objective in the modern Internet. Two main architectures have been standardized by the Internet Engineering Task Force (IETF) for quality of service support: Integrated services architecture (Intserv) and Differentiated services architecture (Diffserv).

As defined by the IETF, the integrated services architecture is an application oriented QoS solution that imposes per-application states within the network based on a match between the bandwidth requested by an application and the response of the network. While being a flexible and powerful approach, the state-full per-application approach does not scale up to large networks.

The differentiated services architecture provides a simple and scalable approach. It is based on the aggregated service concept which allows packets to be classified and marked at the ingress point (the point where the traffic enters the network) based on the information contained in the DS field of the IP packet header to support different service levels and allow these different service levels to be shaped, policed and forwarded differently within a Diffserv domain. Despite its scalability, the Diffserv approach does not provide QoS guarantees to individual applications.

An emerging best-effort architecture based on pricing mechanisms is being extensively investigated by the IP community. It is based on a single class model allowing users to mark packets based on the cost of forwarding packets and uses feedback mechanisms to allow users to adjust their transmission rate to the price of resources. Using this model, differentiation of services will be supported by a market-pricing scheme where resources will be priced by the network based on the load level produced by the traffic in the network and users will implement a willingness-to-pay scheme where resources will be purchased according to the importance accorded (priorities) to the traffic. This model raises the expectation that the modern Internet will still be dominated by

a best-effort traffic but with more control to support congestion avoidance and differentiation of services.

The design of an architecture that offers the best of the three architectures requires a careful investigation of the impact of the different architectural features on different performance and scalability issues involved in the modern Internet:

- Flexibility and scalability are two important features which the different controls deployed for bandwidth allocation and congestion detection in the network depend upon. The localization of these different controls in the network infrastructure is an issue which can affect network performance.
- Methods deployed in routing updates for the determination of the network performance parameters may be time- and resource-consuming. Furthermore these mechanisms might not be able to meet the constraints expressed by QoS routing models.

This chapter proposes a routing architecture to be deployed in path-oriented environments. The remainder of this chapter is organized as follows. Section 4.2 presents the proposed routing architecture. Section 4.3 discusses scalability issues related to our routing choices. The implementation strategy is presented in section 4.4. The conclusions are presented in section 4.5.

4.2 The Proposed Routing Architecture

Most current IP routing architectures that provide QoS implement some form of centralized broker function. These architectures require a signalling protocol to provide an admission policing function plus route computations over a single domain. Emerging architectures are proposing the distribution of the functionality of the single centralized entity across the whole network by replicating or partitioning the broker database among distributed brokers. These architectures require a protocol to maintain the consistency of the broker databases.

The use of a centralized bandwidth broker that maintains the network topology as well as the state of the entire network is more appropriate for long-term traffic variation environments where most flows are long-lived and set-up and tear-down events are rare. In contrast, a distributed broker architecture is well suited to support fine grain and dynamic flows. It is therefore expected that architectures combining the best of the centralized and decentralized broker architectures could provide efficient traffic management and increase the GoS offered by the network. We therefore propose a decentralized routing architecture based on a distributed broker function allowing long- and intermediate-term variations of the traffic to be controlled by decentralized controllers installed at the edge of the network, while short-term variations of the traffic are controlled through a user-core interaction using an end-to-end control model.

4.2.1 The Key Routing Features

The key features of the routing architecture are:

- Distributed bandwidth management by allowing different time scale variations of the traffic to be controlled by different processes from the edge of the network or by the end applications.
- Decentralization of processing mechanisms (network provisioning and path set-up) by either piggy-backing these mechanisms on existing protocols (OSPF, ISIS-ISIS) or by the design of lightweight protocols allowing the mapping of centralized functions into distributed mechanisms.
- Decentralization of the processing information by distributing the information about the topology and routing updates among the local controllers installed at the edge of the network.
- Multi-path routing by allowing multiple routes to be computed between each source-destination pair to install path sets that are downloaded to the transit routers from the ingress nodes.
- Load-balancing of the identified routes by bandwidth controllers installed at ingress routers.
- Use of link controllers installed in the core of the network as link cost processors and markers for packets which cause the link to exceed a defined level.
- Deployment of different control mechanisms for different variations of the traffic allowing the separation of the network into two control planes: a user-user control plane and an edge-edge control plane.
- Long and intermediate-term variations of the traffic involving path set-up and network re-configuration are controlled by the edge-edge control plane while short-term variations of the traffic are controlled by the user-user control plane.
- Hybrid network management supported by the layering of two different control structures above a Diffserv architecture: a mono-layer explicit feedback control model in the user-user signalling control plane and a two-layer hybrid feedback control model in the edge-edge signalling plane.
- The user-user control plane is based on an architecture implementing a user-core interaction where users adjust the transmission rate of the traffic admitted into the network using feedback information received from link controllers installed in the core of the network.
- The edge-edge control plane is based on an architecture layered on top of the Diffserv architecture to control the network using an edge-core interaction allowing global optimization mechanisms to be implemented for computing optimal operational parameters and allowing network re-optimization to be deployed when network parameters have moved from their optimal settings.

4.2.2 The Proposed Network Architecture

We consider a path oriented environment implementing a Diffserv routing architecture allowing the implementation of flexible schemes where resource allocation and congestion detection functions may either be performed at the edge of the network or shared between the edge of the network, the core of the network and users. In the edge-edge control plane, the processing load is moved to the edge of the network while the forwarding process is left in the core of the network and the adjustment of the traffic is carried out by end applications. The user-user control plane shares the processing load related to resource allocation between end-applications and the core of the network. In the path-oriented environment considered, the traffic is engineered based on a routing interaction involving users, core nodes and edge network nodes. The edge routers (LER) are installed at the edge of the network where packets are labelled and conditioned before entering the core of the network. LERs also play the role of distributed bandwidth controllers monitoring paths through local control mechanisms namely dynamic resizing, path set augmentation and diminution. Core routers installed in the core of the network are used to forward packets that have been labelled at the edge of the network and to tag packets that cause congestion to reflect the actual path (network) load. End systems include user and application host systems that generate the traffic sent into the network and adjust the transmission rate of the traffic according to the state of the network.

4.2.3 The Edge-Edge Control Plane

Key features

The key features implemented in the **edge-edge control plane** are:

- Functional separation between traffic transmission and traffic measurements through the use of two functional subnetworks: user/edge and edge/edge subnetworks.
- Implementation by the user/edge subnetwork of an implicit feedback model based on a lightweight protocol allowing users to signal their bandwidth requests to the edge of the network and the edge to feedback control information to users to adapt their transmission rates to the received information.
- Implementation by the edge/edge subnetwork of a multi-path routing model based on three functions: computation of path sets, load-balancing of traffic between available paths at the edge of the network, and packet forwarding and cost (load) measurements in the core of the network.
- The feedback control within the edge-edge control plane is performed using an explicit feedback scheme where costs computed by core controllers are fed to the ingress nodes which distribute the network bandwidth accordingly.

- Periodic control by edge controllers of the allocation of the traffic offered to path sets and periodic updates of a database containing the statistics concerning the use of their paths. This database is called the label statistics base (LSB).
- Triggered re-computation of optimal path sets by the edge controllers using a set-up/reconfiguration algorithm that uses the information provided by the distributed database installed at the edge of the network.
- Implementation of a hash-threshold at the edge of the network to avoid packet re-ordering.

The Network Control Model

The edge-edge control plane includes local controllers installed at the edge of the network to control global network provisioning and reconfiguration mechanisms through the implementation of lightweight protocols to map the centralized processing required for global control into local controls and the decentralization of the storage of information.

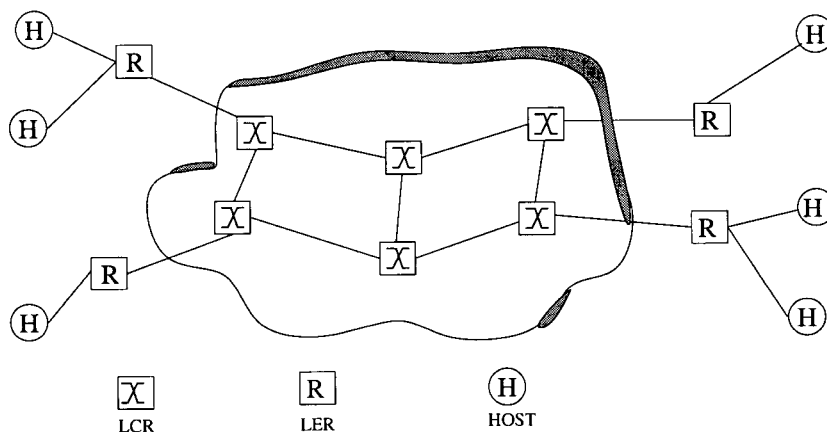


Figure 4.1: The “edge-edge” plane

We propose a two-layer network architecture illustrated by Figure 4.1 where user-edge subnetworks composed of Hosts and Label Edge Routers (LER) implementing the traffic adjustment are layered above an edge-edge subnetwork which controls the load-balancing, cost computation and packet forwarding. The edge-edge subnetwork include the LERs and the Label Core Routers (LCRs). In this architecture, the edge-edge subnetwork implements an explicit feedback scheme where the traffic is shared among existing paths at the edge of the network based on periodic path cost measurements collected in the core of the network. The user-edge subnetwork implements an implicit feedback scheme consisting in adjusting the traffic based on a binary feedback received from the edge of the network which controls the network load and resource allocation.

We consider a network management model where the control of the traffic entering the network is done at the edge of the network (edge-based model) by load-balancing the traffic over the available paths. This edge-based model is a multi-path routing model where the traffic transmitted by end applications is regulated using four mechanisms: path set augmentation, load-balancing of the traffic over available paths, hash-based routing to avoid packet reordering and aggregated feedback.

Path set augmentation

The routing model considered is based on a traffic distribution starting with a minimum number of paths and progressively increasing the number of paths when the path set bandwidth reaches a load level expressing an infeasible flow or reaches a congestion avoidance threshold. Load levels used in this model may have several semantics: utilization, slack, etc. We consider a model where the load level is represented by either the path set bandwidth, the path set utilization, the path set threshold or any other loading measure presented in chapter 2. Path set augmentation is triggered by a periodic signalling process which triggers new path activation based on the load measurements received from the core of the network. This provides a form of dynamic path set over-provisioning by activating back-up paths to share the traffic offered to the path set when required.

Load-balancing

The edge-based model pre-computes several paths for each source-destination pair. These paths are load-balanced based on the contents of a flow statistics database containing for each path its actual cost. The link costs and utilization are computed by core link controllers and periodically advertised to the edge of the network.

Aggregated feedback

The routing approach implements a congestion avoidance mechanism based on the activation of back-up paths and a feedback control which is triggered when the edge controllers have activated all their available back-up paths and need additional bandwidth to carry the offered traffic to avoid infeasible flows. After activation of all the back-up paths available in a path set, a feedback notification is sent to all traffic requirements using that path set to adjust their transmission rate to the available bandwidth. We consider an aggregated feedback scheme where the same feedback signal is broadcast to all sources transmitting their traffic using the same path set.

Packet Reordering

When deployed naively, for example forwarding each packet using the next hop in a round-robin fashion, multi-path routing may lead to packet re-ordering due to the presence of multiple paths with different latencies. Packet re-ordering causes TCP to enter the “fast re-transmit” mode where

extra bandwidth is consumed in unnecessarily re-transmitting packets which have been delayed due to packet disordering. We consider a routing model where packet reordering is avoided by the implementation of the hash-based routing method presented in the following section.

4.2.4 The User-User Control Plane

Key Features

The key features implemented in the **user-user control plane** are:

- Implementation of a mono-layer network architecture based on an explicit feedback scheme where the costs computed by core controllers are fed to the end applications which adjust their transmission rate accordingly.
- Triggered routing updates which adjust the traffic on a path when the path load has reached a predefined threshold or congestion level.
- Single-path routing based on an end-to-end routing model where the traffic transmitted in the network is adjusted by end applications: little participation is required from ingress routers which participate only in the route computation process.
- The distribution of the network bandwidth to the traffic is controlled by the end users which decide on the quantity of traffic to be admit to the network based on the feedback received from core controllers.

The Network Control Model

The user-user control plane implements traffic regulation mechanisms based on a user-core interaction where end applications adjust their transmission rate based on feedback of link controllers expressing the load levels reached by network links. This is achieved by the implementation of explicit feedback mechanisms in the user-core subnetwork allowing users to adjust their transmission rate based on explicit notifications of the core (link controllers).

4.2.5 The Proposed Edge Router Architecture

This section presents a label edge router (LER) architecture based on enhancing the MPLS label distribution and flow classification with new functionalities and by enriching the MPLS label information base (LIB) with new components. These new components illustrated by Figure 4.2 include a packet forwarding component called the packet forwarding base (PFB) and a LSP statistics database called the LSP statistics base (LSB). The PFB, the LIB and LSB form an enhanced database called the flow information base (FIB).

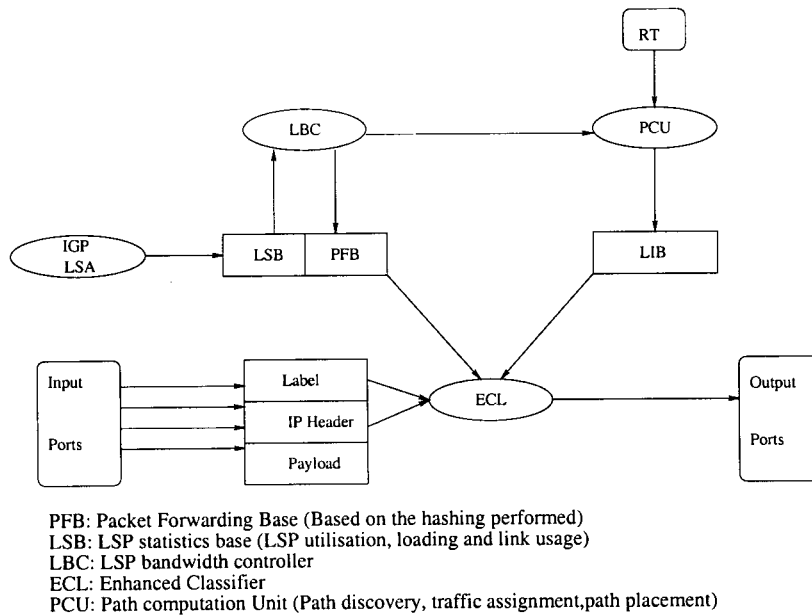


Figure 4.2: The “LER” router

The Packet Forwarding Base

The PFB is constructed based on the hashing performed during the LSP set-up process. It includes a set of forwarding entries for each ingress node which supports multi-path routing. Forwarding entry sets are attached to LIB entries for which the hashing has been performed. In the case of congestion the contents of the PFB are temporarily adjusted to provide alternate paths for congested traffic.

The LSP Statistics Base

The LSB contains information concerning the use of the LSPs. Each LSB entry contains the corresponding LSP utilization rate, its loading, its maximal link interference given by the maximum over its links of the number of LSPs which use the link, its cost defined as in chapter 2 and other measures.

The LSP Bandwidth Controller

An LSP bandwidth controller (LBC) is installed in each network edge node which supports multi-path routing to control the bandwidth allocation of its LSP sets. Periodically or based on a trigger the following tasks are performed by the LBC:

- Collection of statistics related to the use of the links. Such statistics are provided by link

state protocols such as OSPF-LSA and are used to measure the use of LSPs and maintain the LSB.

- Trigger new LSP set discovery by instructing the path computation unit (PCU – see below) to compute new LSP sets for the LIB and new hashing tables for the PFB. New LSP set discovery is performed when a persistent congestion of the LSP sets is observed.
- Adjust the PFB content when an LSP set is experiencing random mismatches between its offered traffic and the LSP set bandwidth. Such adjustment is applied only to the LSB contents. No new LSP set computation is performed.

The Enhanced MPLS Classifier

The ECL forwards packets according to the information contained in the flow information base (FIB) and the MPLS packet headers (also called the shim header).

The Path Computation Unit

The PCU is the routing component of the architecture while the ECL constitutes its forwarding component. The PCU installs the initial LSP sets and packet forwarding sets in the FIB. It also computes new LSP sets based on a bandwidth controller trigger when persistent congestion occurs on some or most of the LSP sets.

4.3 Scalability Issues

4.3.1 Decentralized Routing

Communication network architectures have traditionally been dominated by two approaches: centralized approaches where bandwidth allocation and routing functions are controlled by a central observer/controller node, and decentralized consensus-based approaches where resource allocations and routing processes are conducted by consensus between network nodes. While being simple, the centralized approach requires frequent transmission of signalling information from all the nodes of the network to the central observer in order to adjust the network configuration to changing traffic patterns. This consumes a good portion of the network bandwidth that should be used to carry data information. Furthermore, the central observer approach does not provide any network availability when the central observer fails.

The decentralized consensus approach is more reliable than the central observer approach. However, this approach also transmits considerable signalling information among network nodes and does not cope with the multiplicity of factors involved in modern network technologies. These factors include selfish behaviour of users competing for resources, multiplicity of inter-working protocols and service classes, and stringent requirements.

Besides the need to provide real-time traffic management, routing architectures deployed in the modern Internet must adapt to the nature of modern Internet protocols including TCP, RED, SRED which tend to move the control of the network to the edge of the IP domain by deploying more control at the reception and departure of IP packets. Moving the control of the network to the periphery of the network results in overheads in links installed at the edge of the IP domain. Achieving real-time traffic management by performing different time scale control mechanisms to the edge of the network by deploying a decentralized consensus approach may result in decreased performance by moving more load to the edge of the network and increasing the network signalling. The deployment of a central observer approach is neither natural due to the decentralized nature of modern networks nor efficient since it requires excessive signalling. A decentralized architecture where all processing (short-, intermediate- and long-term) is controlled by the edge of the network is more scalable.

We consider a model where the centralized operation is mapped into decentralized mechanisms through the implementation of lightweight protocols allowing centralized operations to be distributed among edge nodes where local operations are executed. This model provides several advantages compared to centralized models including less complexity at the edge of the network, less processing power required from LER installed at the edge of the network, less signalling overhead required for route computation, less traffic allocation and routing updates, simplified administration provided by centralization at the edge, and robustness through decentralization.

4.3.2 Multi-path Routing

Routing in IP networks uses two approaches : single path routing and multi-path routing. In single path routing, data packets are sent from a source to a destination using a single path. Multi-path routing exploits the connectivity of the underlying physical network by sending successive packets of the same flow among multiple paths. Multi-path routing presents some advantages over single path routing. It provides greater end-to-end throughput and better QoS by allowing the traffic offered to an O-D pair to be split among multiple paths and load balancing the available paths to increase the network utilization.

We consider a routing architecture that exploits multi-path routing for traffic engineering and differentiated services to increase the quality of service offered by MPLS networks.

Multi-path routing schemes include a path computation process which discovers a set of paths with sufficient spare capacity to route the traffic, and a selective traffic distribution process which selects the best among the set of paths found to carry the offered traffic. The best path is generally selected

according to a simple objective such as delay minimization or spare capacity or according to multiple objectives combining two or more simple objectives. Two main approaches are commonly used to compute feasible paths: reactive schemes compute paths simultaneously with the traffic distribution and pre-planned schemes pre-compute paths before any traffic distribution. Pre-planned control schemes are usually deployed in networks where traffic patterns do not change frequently. Reactive control schemes are more suitable for networks with rapid change of traffic patterns and where information updating and route pre-computing would become a burden. We consider an hybrid approach consisting of the pre-computation of paths and the dynamic activation of these paths according to the network load. This model presents the advantage of reducing the complexity of path re-computation involved in reactive schemes and distributing the traffic to the minimum number of paths which are activated on-demand.

4.3.3 Packet Reordering

Three hashing methods were proposed in (Thaler and Hopps, 1999) for improving the performance of multi-path routing for faster implementation and minimal disruption. These methods are based on applying a hash function to the packet header fields that identify the traffic flows and forwarding the packet according to the hashing performed. They include:

- **Modulo-N Hash** performs a modulo- N hash over the packet header fields that identify a flow. This allows the selection of a next-hop from the list of N next-hops. This fast method only has an overhead of $(N - 1)/N$ of all flows changing paths whenever a next-hop is added or removed.
- **Highest Random Weight** computes a key for each next-hop by performing a hash over the packet header fields that identify the flow, as well as over the address of the next-hop. Thereafter, the highest resulting key value is selected as the next-hop. Highest Random Weight has the advantage of minimizing the number of flows affected by a next-hop addition or deletion but is approximately N times as expensive as modulo- N hash.
- **Hash-Threshold** initially selects a key by performing a hash (modulo- K where K is large, or CRC16) over the packet header fields that identify the flow and assigning unique regions in the key space to the N next-hops. Thereafter, the key is compared against region boundaries to determine which region the key belongs to and which next-hop to use. Hash-threshold has the advantage of only affecting flows near the region boundaries (or thresholds) when next-hops are added or removed. It has been reported that when a next-hop is added or removed, between $1/4$ and $1/2$ of all flows change paths.

4.3.4 Traffic Measurement

Some of the traffic measurement metrics that have been used in traffic engineering are:

- *Packet Delay and delay variation:* In our scheme, packet delays on paths are obtained by transmitting probe packets from the ingress node to the egress node. This is achieved by time-stamping the probe packet at the ingress node at the time of transmission and recording the time of reception of the probe packet at the egress node. In this model, the problem of clock synchronization between the routers installed at the ingress and egress nodes is solved by adding the difference of clock rates to the recorded round-trip time of the probe packet.
- *Packet Loss Probability:* The packet loss probability may be estimated by sending a set of probe packets from the ingress to the egress node. This is done by encoding a sequence number in the probe packet to notify the egress node how many probe packets have been transmitted by the ingress node and another field in the probe packet to indicate how many probe packets have been received by the egress node. The packet loss probability is estimated by comparing the number of probe packets that have been transmitted to the number that have been received.
- *Available Bandwidth:* Different approaches have been used for measuring the available bandwidth of a given resource. The routing architecture presented above allows the application of a simple approach where the available bandwidth of a given resource is recorded during routing updates. This measure is used to detect congestion in the network and trigger appropriate mechanisms.

4.3.5 Routing Updates

QoS routing support may impose stringent demands on the network to express the bandwidth and processing load required from each router (switch) involved in the routing process for maintaining its view of the available resources, to exchange the routing information with other routers (switches) and recompute routes which meet the defined QoS requirements. The accuracy of the routing decision taken by a router depends on its current view of the bandwidth available on all links in the network which also depends on the type of the routing update algorithm used and the frequency of the update messages.

Periodic and triggered updates are the two most deployed routing update methods. Periodic updates are based on a periodic invocation of routing updates while triggered updates propagate the link-state information in response to a trigger expressing either a significant change in the link-state metric considered or a defined threshold. Periodic and triggered updates involve bandwidth and processing load to allow link-state information to be propagated.

Tuning the frequency of link-state update messages leads to a tradeoff between the network processing overheads and the performance gains resulting from the routing decisions involved in the link-state information propagation. While giving a more accurate view of the bandwidth available on links, frequent updates called each time the measured link metric changes are neither

scalable nor practical. Infrequent updates based on a maximum spacing between routing updates may lead to inaccurate routing decisions.

Periodic updates provide the advantage of reliable and fine-tuned link state updates. Furthermore, when deployed in a slow time-varying traffic environment, routers involved in periodic updates may implement a learning procedure where the interval between link-state updates is adjusted to the traffic profile expressed for example by the time-zone variations or the network load expressed by the signalling overhead experienced by the network. Triggered updates are the best alternative for traffic bursts and random mismatches between the offered traffic and the available bandwidth.

We consider a routing model implementing periodic updates in the edge-edge control plane where the traffic is assumed to present a slow time-variation profile. This model is complemented by the deployment of triggered updates in the user-user control plane where link-state advertisements are triggered when the link state has reached a defined threshold or congestion level.

The learning procedure referred to above is based on the belief that the network bandwidth may be distributed freely among traffic requirements under light load conditions and “preferentially” shared between requirements under heavy load conditions. It is therefore expected that a large interval between routing updates be deployed between routing updates under light load and a small interval be deployed under heavy load conditions. This learning procedure may adjust the interval between link-state updates based either on the offered traffic expressed by the network signalling overhead or by following a traffic profile as presented in Figure 4.3.

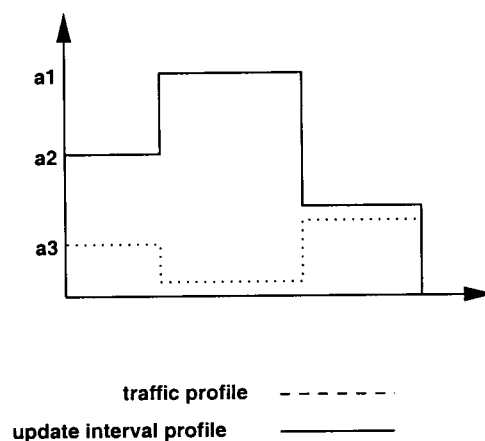


Figure 4.3: The “learning” model

The interval between routing updates may be expressed by a distance function:

$$D_t = f(a) \quad (38)$$

where a is the total traffic offered to the network. It is expected that a learning procedure based on a function D_t following a piece-wise variation of the traffic with respect to the time will produce the routing updates interval profile presented in Figure 4.3.

It can be observed that the routing update interval profile follows the traffic profile by increasing the interval when the traffic decreases and by decreasing the distance when the offered traffic increases.

4.4 Implementation Strategy

Two main implementation issues are involved in the implementation of the routing architecture in a path-oriented environment: how traffic measurements are performed by the core of the network and how feedback mechanisms signal the network load to end applications.

4.4.1 Traffic Measurement

Traffic measurements are important for any traffic engineering approach. Different traffic measurements may be performed by the core and edge nodes: Packet delay and packet loss probability measurements are triggered by the edge routers while link costs are measured by core routers. End applications do not perform any traffic measurement. We consider a traffic engineering approach where the sum of link costs and the maximum link utilization rate are used in the edge-edge control plane to control the load-balancing at the edge of the network and detect congestion within path sets. In the user-user control plane, link utilization rates are mapped into marks which are to be used as cost in the adjustment of the transmission rate of the traffic. This differs from common IP traffic measurement practices where the packet loss probability is used as marker for packets which produce losses in queues.

4.4.2 The Edge-Edge Control Plane

We consider a routing model where the cost computed by link controllers consists in the derivative of the link delay with respect to traffic flow and the maximum link utilization expresses the loading level reached by the network. In this explicit routing model, the path cost is measured by summing the link costs and computing the maximum link utilization. The signalling model consists in periodically sending probe signals from a source to a destination to collect the sum of link costs and the maximum link utilization rate. On its trip to the destination, this signal computes at each link the sum of the link costs and the maximum link utilization which, upon arrival at the destination, are fed back to the ingress router which updates its label statistics base (LSB). This is illustrated in Figure 4.4.

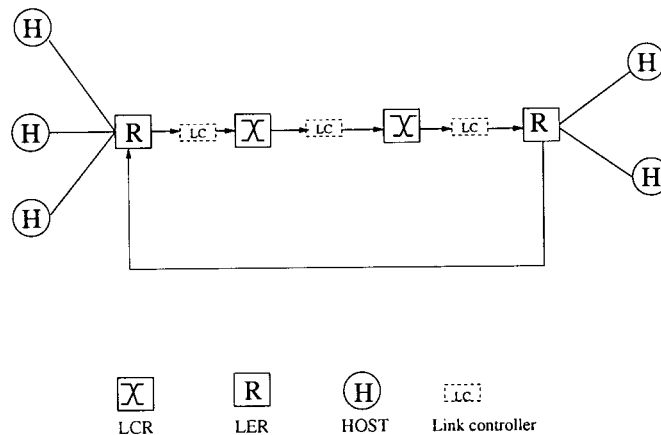


Figure 4.4: The “edge-edge feedback” model

The User/Edge Signalling Model

The implementation adopted in this work considers a distributed broker function controlled by edge controllers which play the role of load-balancers at the ingress of the network. In this architecture, binary control values are computed based on the congestion level reached by the network and advertised through a lightweight protocol that allows users to signal their bandwidth requests to the ingress of the network and the core to feedback load information to allow users to adapt their transmission rate. A two-bit signalling scheme is considered where one bit is used to advertise the type of traffic adjustment (increase or decrease) and another bit is used to advertise the type of decrease (additive or multiplicative). The signalling protocol includes an edge and a core protocol.

The Edge/Edge Signalling Model

Path cost measurements are based on periodic transmissions of signalling messages carrying the sum of the costs of the links traversed by the paths and the maximum of their link utilization. Upon reception of the message, each link controller localized along the path computes its own link cost and link utilization which are respectively added and compared to the values carried by the signalling packet. Thereafter, the maximum link utilization carried by the message is set to the value of the link utilization if the link utilization is greater than the maximum and the signalling message is forwarded to the next hop along the path to the destination where a similar algorithm is implemented. Upon arrival at the destination, a new signalling message containing the cost and utilization rates computed is sent to the ingress router where the LSB is updated accordingly and either path set augmentation or feedback signalling is performed.

We consider a marking model where the cost computed by link controllers consists in marks representing the link utilization rate expressing the loading level reached by the network. The signalling model consists in marking packets in the core of the network and either summing or

computing the maximum over link's marks. In this explicit routing model, the path cost is measured by either summing marks allocated to the traffic flowing through a path or the maximum over marks computed on links traversed by that path.

Routing Updates

In the edge-edge control plane, periodic routing updates based on a learning procedure are implemented which adjust the interval between routing updates according to the offered traffic. The interval between routing updates is computed based on the traffic offered following a function $D_t = f(a)$ assigning to each value of the traffic a a different value of the routing update interval x . It can be noted that high values of the traffic a expressing a heavy load should naturally lead to a short spacing between routing updates while low values of the traffic a expressing light load conditions should lead to a large interval between routing updates.

4.4.3 The User-User Control Plane

We consider an additive model where, upon reception of the packet, a core router adds the link mark to the marking field of the packet and forwards the packet to the next-hop router which executes the same actions. Upon reception of the marked packet by the receiver, a new packet containing the collected information is returned to the sender which adjusts its transmission rate accordingly. A signalling based on the additive model presented above is different from a maximum path cost signalling model where the maximum over the link marks are forwarded to the next-hop router instead of the sum over the link marks. Additive marking requires an entire packet to carry the computed sum while maximum marking requires only a few bits to convey the values of the marks.

Routing Updates

The triggered routing update model provides a natural way for regulating short-term varying traffic resulting from random mismatches between the offered traffic and the available bandwidth on paths. In the user-user control plane, routing updates will be triggered when the link utilization has reached a threshold region or a congestion region where the link load exceeds its available capacity.

4.5 Conclusion

This chapter presents a routing architecture allowing the traffic offered to the network to be regulated based on a signalling model using two control planes: an edge-edge control plane and a user-user control plane. The edge-edge architecture implements a distributed model allowing

network provisioning and reconfiguration mechanisms deployed on a long-term time scale of the traffic variations to be controlled by local controllers installed at the edge of the network. The user-user control plane implements an end-to-end signalling model supporting the control of short-term variations of the traffic.

Scalability issues related to this routing architecture have been discussed. While supporting the sizing and resizing of network paths according to the offered traffic based an end-to-end load-balancing scheme, the edge-edge architecture presented does not produce scalability issues related to the installation of a per-flow state in the core of the network: all path sizing and resizing information is kept only at the ingress of the network.

The focus on this chapter lies on the signalling model of the routing process and the interplay between link-state update policies, traffic patterns and the performance gains resulting from these update policies.

Periodic routing updates implemented in the edge-edge control plane are complemented by triggered controls deployed in the user-user control plane to implement an effective regulation of the traffic entering the network.

The signalling model presented may be either piggy-backed on network administrative tools such as SNMP and MIB used in remote monitoring of routers or piggy-backed on existing protocols such as those presented in the RFC2676 which deals with QoS routing mechanisms and OSPF extensions. The signalling model may also be implemented as a protocol aiming at providing the required functionality for Internet application users to capture the overall network performance.

Network performance parameters can be estimated either by changing or without having to change the software in the routers involved. The signalling costs involved in the two alternatives and their relative performance gains need to be compared in order to assess the relative merits of these alternatives. This is reserved for future research work.

The performance of the proposed routing architecture is evaluated in chapter 5.

Chapter 5

Traffic Regulation

5.1 Introduction

The evolution of the Internet from a non-cooperative network into a commercial platform requiring more than the GoS offered by the traditional best-effort service deployed for data traffic leads to a tradeoff between the economic ramifications of the bandwidth sharing policy deployed in the network and the engineering efficiency achieved by transmission rate allocation algorithms. It is believed (Gevros et al., 2001) that the modern Internet will still be dominated by best effort traffic but with more control to prevent congestion collapse, to achieve low congestion levels and guarantee fairness to the best effort subnetwork. How the traffic is allocated to different paths in the best effort subnetwork is therefore an important aspect that affects the performance of traffic engineering schemes and the QoS achieved by the network.

Feedback mechanisms used in congestion avoidance schemes are classified into two categories which are referred to as explicit feedback and implicit feedback schemes. Implicit feedback schemes also called bit-based feedback schemes are based on a binary indication of congestion issued by the network to allow users to adapt their transmission rate to an estimation of the network load. These schemes require little processing from the network: little participation is required from core routers and switches and few exchanges of information between the network and users/applications. Explicit feedback schemes involve a distributed computation of transmission rates where the transmission rates are computed by the network and sent to users/applications as feedback information to adapt their transmission rate to an estimation of the network load. Though moving the processing overhead towards the network, this scheme achieves performance properties such as efficiency, fairness, controlled queueing delay and robustness. The integration of transmission rate allocation schemes based on feedback mechanisms in the existing and emerging routing architectures raises issues related to the localization of the processing entities in the routing environment and the nature of the signalling protocols implementing feedback mechanisms.

A flow optimization model based on an incremental procedure allowing the traffic offered to a network to be allocated in small fractions of the offered traffic was presented in chapter 2. One of the main objectives of this model and the subsequent allocation of the traffic to precomputed paths in small increments is to provide a flow allocation scheme that may be deployed in off-line as well as in on-line routing environments with few signalling adjustments and improved performance. Indeed, while being evaluated in an off-line environment using a scheme where the traffic increment granularity is at the level of the aggregated traffic, the flow allocation model implements a traffic regulation scheme similar to congestion avoidance mechanisms deployed in on-line environments (Jain and Chiu, 1989) where the transmission of the traffic into the network is regulated by a feedback mechanism allowing the offered traffic to be allocated based on an estimation of the network load.

Several issues related to the implementation of this flow optimization model in an on-line environment and other issues related to the regulation of traffic in path-oriented schemes are investigated in this chapter:

- The need for QoS in IP networks has recently resulted in a great interest for mechanisms supporting the implementation of differentiated services in IP networks. How differentiated services are implemented in an environment which routes the traffic in small increments is an issue to be investigated.
- The flow optimization model presented in chapter 2 was evaluated by load-balancing the traffic over multiple paths computed for each source-destination pair. Mono-path routing schemes have been deployed in the Internet using different adaptive rate algorithms to achieve a network optimality based on a fairness criterion. How load-balancing schemes compare to adaptive rate algorithms and achieve network fairness while guaranteeing operational requirements is another issue to be investigated.
- Congestion avoidance mechanisms deployed in mono-path routing schemes are based on the decrease of the transmission rate of the traffic sent over these paths. Multi-path routing schemes allow an alternative form of congestion avoidance consisting in routing the traffic over more paths when the traffic can not be accommodated by the paths which currently carry the traffic. Allowing traffic regulation mechanisms deployed in multi-path routing environments to take advantage of this form of congestion avoidance may change the way feedback control mechanisms are implemented.

This chapter evaluates through simulation the routing architecture proposed in chapter 4. We propose a multi-path routing approach for IP traffic regulation in a path-oriented environment where the traffic offered to the network is distributed among multiple paths by bandwidth controllers installed at the edge of the network using a bandwidth negotiation protocol aiming at effecting the best tradeoff between the available network bandwidth, the application bandwidth requirements and the network performance requirements. The fairness achieved by this approach

is evaluated by comparing the throughput produced by UDP flows using this approach compared to a manually configurable explicit routing scheme and a best effort scheme. Differentiated services are supported by assigning different traffic increments to different traffic flows according to their priority.

The remainder of this chapter is organized as follows. Section 5.2 presents the traffic regulation model. Service models are presented in section 5.3. Section 5.4 discusses the network fairness achieved by traffic regulation schemes implementing mono- and multi-path routing. Section 5.5 presents simulation analysis. The conclusions are presented in section 5.6.

5.2 The Traffic Regulation Model

Feedback schemes for transmission rate and congestion avoidance were investigated in the past (Jain and Chiu, 1989) and a proposal allowing a bit in the packet header called the congestion experienced bit to be used for feeding back congestion notification to users/applications was incorporated into the Open System Interconnection (OSI) connection-less network protocol standards (OSI, 1986). Furthermore, the well-known linear increase/exponential decrease transmission rate model implemented in the TCP congestion control algorithms was motivated by the additive increase multiplicative decrease (AIMD) initially proposed in (Jain and Chiu, 1989).

Feedback controls involve two main functions: a congestion detection and a resource allocation function. Congestion detection functions are binary functions that are computed by the network by comparing the the demand for resources with their availability to detect any mismatch expressing an under-load or an over-load. Resource allocation functions compute the share of a resource that is allocated to user/applications based on a tradeoff between the available resources, the demand for those resources and the routing policy deployed. This section presents the feedback model to be deployed in the routing architecture presented in chapter 4.

5.2.1 Traffic Regulation Methods

Path oriented environments implementing a Diffserv routing architecture allow the implementation of flexible feedback schemes whereby the resource allocation and congestion detection functions may either be performed at the edge of the network or shared between the edge of the network and the core of the network. While moving the processing load to the edge of the network, this model leaves the forwarding process in the core of the network and the transmission of the traffic to users/applications. We consider a routing model where the cost computed by link controllers consists of the link derivative delay and the maximum link utilization expressing the load level reached by the network. In this explicit routing model, the path cost is measured by summing the link costs and computing the maximum link utilization as proposed in chapter 4.

Two traffic regulation methods are proposed:

1. a traffic regulation model combining dynamic path activation with the use of an additive or a multiplicative decrease feedback model, and
2. a model implementing an hybrid threshold feedback model over pre-selected paths using a static path activation model.

The first model activates additional paths to route the traffic under congestion or reduces the transmission rate of the traffic if there are no more back-up paths to be activated. The second model distributes the traffic among preselected paths based on the sum of link costs and reduces the transmission rate of the traffic when the paths carrying the traffic have reached a defined path set threshold.

It can be observed that:

- The two proposed models include a form of congestion avoidance allowing the traffic to be routed over more paths instead of having the sources to reduce their transmission rate.
- The dynamic path activation proposed in the first model is a form of dynamic network over-provisioning allowing the offered traffic to be accommodated due to the increase of the number of paths available in a path set.
- The first model presents the advantage of routing the traffic over a limited number of routes since new routes are activated only when required. However it requires an extra protocol for path activation. The second approach does not require any additional protocol for path activation but may distribute the traffic to more paths. This can be in conflict with operational requirements which sometimes require the traffic to be carried over a minimum number of paths.

We propose a traffic regulation model where thresholds expressing the maximum link utilization levels reached by paths are used as triggers to either adjust the transmission rate of the traffic or to increase the number of paths available for routing the traffic.

Path cost measurements are based on periodic transmissions of a signalling message carrying the sum of the costs of links traversed by the paths and the maximum of their link utilizations. Upon reception of that message, any link controller localized along the path computes its own link cost and link utilization which are respectively added and compared to the values carried by the signalling packet. Thereafter, the maximum link utilization carried by the network is set to the value of the link utilization if the link utilization is larger than the maximum and the signalling message is forwarded to the next hop along the path to the destination where a similar algorithm is implemented. Upon arrival to the destination, a new signalling message containing the cost and utilization computed is sent to the ingress router where the LSB is updated accordingly and either path set augmentation or feedback signalling is performed.

5.2.2 Routing Functions

We consider a path-oriented environment where the traffic is engineered based on a routing interaction involving users, core nodes and edge network nodes. In this model, the user implements a traffic adjustment function, the core performs packet forwarding and traffic measurement functions and the edge router implements three functions: congestion detection, routing updates and load-balancing.

Ingress Algorithm

After each time interval ΔT^e , the ingress:

- initializes signalling packets containing a sum field and a maximum field where the sum field representing the sum of link costs and the maximum field representing the maximum link utilization are set to 0.
- forwards these packets into the network along paths to collect the path cost and its utilization.

Three functions are implemented by the ingress of the network: feasible flow computation (congestion detection algorithm), routing updates and the load-balancing.

1. Congestion Detection Algorithm

The load level reached by a path set is expressed by its path set threshold defined by the number of paths which have reached the threshold load. The congestion detection algorithm implemented by the ingress router consists of computing the path set threshold and comparing this number with the number of active paths to detect an overload expressed by a majority of paths under threshold load or a moderate load when the path set threshold is low.

A high level description of a congestion detection algorithm using a path set threshold is as follows:

- compute the path set threshold $T_{i,e}$, and
- signal congestion if $T_{i,e} \geq |\mathcal{A}_{i,e}/2|$

where $\mathcal{A}_{i,e}$ is the active set of the path set $\mathcal{P}_{i,e}$ and $T_{i,e}$ is the path set threshold.

2. Label Statistics Updates

Routing updates are based on the information collected in the label statistics base (LSB). This information consists of path costs and maximum link utilizations computed by link

controllers in the core of the network. After computation by link controllers, these values are forwarded to the egress of the network and fed back to the ingress by the egress routers.

3. Load-balancing Algorithm

Three algorithms may be implemented by the ingress router: an increase version of the flow deviation algorithm (FLD_i), a decrease version of the flow deviation (FLD_d) and the Weighted Traffic Distribution (WTD) algorithm.

FLD_i/FLD_d/WTD algorithms

- (a) from time to time, the ingress receives path costs and link load levels fed back from links in its paths to the egress of the network and distributes subsequent traffic among paths following the $FLD_i/FLD_d/WTD$ traffic traffic distribution methods presented in chapter 2.
- (b) after each time interval ΔT^e , the edge
 - activates a new active path if the path set threshold exceeds the number of paths below threshold level, and
 - sends a congestion indication to users to adjust their transmission rates if there is no more path to activate.

User Algorithm

- In the absence of negative feedback, the user implements an additive feedback control following the expression:

$$x_r(t+1) = x_r(t) + a$$

where a is a traffic increment, $x_r(t+1)$ is the rate allocated to path r at time $t+1$ and $x_r(t)$ is the rate allocated to path r at time t .

- from time to time, the user receives a two-bit feedback signal (one bit representing the feedback control function $E_r(t)$ and the other representing the feedback control function $L_r(t)$) from the edge. The user interprets the feedback signal as follow:

$$E_r(t) = \begin{cases} 0 & : \text{increase}(x_r(t)) \\ 1 & : \text{decrease}(x_r(t)) \end{cases}$$

$$L_r(t) = \begin{cases} 0 & : \text{slow}(x_r(t)) \\ 1 & : \text{fast}(x_r(t)) \end{cases}$$

where $\text{slow}(x_r(t))$ and $\text{fast}(x_r(t))$ refer to a slow adjustment of the traffic to effect an early congestion detection or a fast adjustment aiming at recovering from a congestion.

5.3 Service Models

- the user adapts its transmission rate according to its interpretation of the feedback signal above according to the following algorithm:

$$x_r(t+1) = (x_r(t) + a) + bE_r(t)L_r(t)(1 - x_r(t)) - E_r(t)(a + b).$$

Core Algorithm

From time to time, a link controller receives a signalling packet from its downstream neighbour. It computes its link cost and link utilization and adds the link cost to the value of the path cost carried by the signalling packet and compares the value of the maximum link utilization carried by the signalling packet with its computed utilization rate to replace the value carried by the packet by its value if its value is larger. Thereafter, the signalling packet is forwarded to its upstream neighbour along the path to the egress which executes the same operation.

The core algorithm implements the following steps:

1. receive a signalling packet carrying the path cost L_p and maximum utilization U_p
2. compute the link cost L_ℓ
3. add the link cost L_ℓ to the sum L_p carried by the signalling packet:
 $L_P := L_p + L_\ell$
4. compute the link utilization U_ℓ
5. if $U_p \leq U_\ell$ then $U_p := U_\ell$
6. forward the packet to its next hop to the egress

Egress algorithm

Upon reception of a signalling message packet, the egress:

- initializes a new signalling packet with the values of the sum of link costs and the maximum link utilization rate
- send the packet to the ingress router for routing updates

5.3 Service Models

5.3.1 Equal Services

The analysis presented in chapter 2 was based on a static equal-service model where traffic requirements are assigned the same fixed traffic increment. A dynamic model where the traffic is

dynamically adjusted by applying the the increase/decrease control functions presented in (Jain and Chiu, 1989) may result in improved performance. Several mechanisms may be used to trigger the adjustment of the traffic increments including the dropping of packets in queues and the use of threshold resource utilization. We consider an approach where the traffic is decreased when network paths reach a threshold utilization.

5.3.2 Differentiated Services

We consider two approaches for service differentiation: static and dynamic service differentiation. In static service differentiation, different traffic increments are applied to different traffic requirements during the increase/decrease operations. These values are unchanged during the routing process. In dynamic service differentiation, different traffic increments are applied to different traffic requirements upon reaching the given load level.

Dynamic service discrimination

The choice of a dynamic traffic assignment is justified and illustrated by Figure 5.1 showing how the bandwidth of a bottleneck link R_3 is shared between two competing flows originating from the hosts h_1 and h_2 .

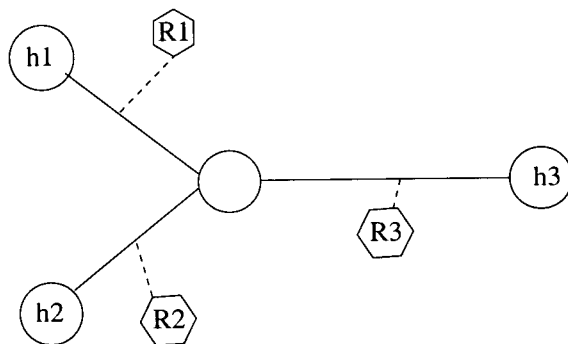


Figure 5.1: The “bottleneck” model

Consider two traffic flows f_1 and f_2 transmitting packets from two hosts h_1 and h_2 to access a server h_3 . These two flows sharing the bottleneck link R_3 are transmitting a total traffic $T = f_1 + f_2$ on that link. Assuming that the capacity of link R_3 is C , the derivative of the delay with respect to flow expressing the link cost if modelled as an $M/M/1$ queue is given by:

$$D(x) = \frac{C}{(C - T)^2} \quad (39)$$

Assuming that the flow f_1 can be expressed as a portion of the flow f_2 using the expression:

$$f_1 = x f_2 \quad (40)$$

5.3 Service Models

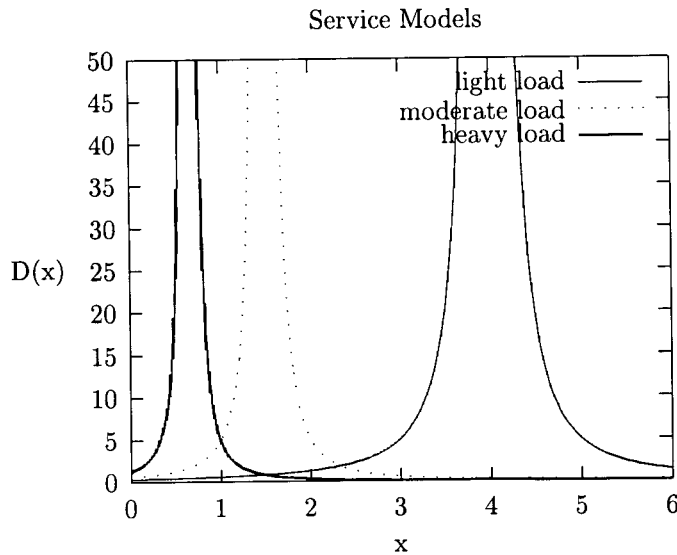


Figure 5.2: Service under different light and heavy load conditions

$$T = (x + 1)f_2 \quad (41)$$

The link cost expression will be rewritten by the following expression:

$$D(x) = \frac{C}{(C - (x + 1)f_2)^2} \quad (42)$$

A graphical representation of this delay function using different load conditions is presented in Figure 5.3.2.

The different load conditions are represented by different values of the flow f_2 in equation (42). We consider the following values: $C = 4$, $f_2 = 0.5$ for light load, $f_2 = 1$ for moderate load and $f_2 = 2$ for heavy load conditions.

It can be observed that:

- an equal service model $x = 1$ gives the lowest delay under light and moderate load conditions
- a differentiated service model performs better in terms of delay under heavy load conditions

The Traffic Prioritization Model

QoS objectives are achieved in an IP environment by providing different types of service to different traffic flows according to their QoS requirements. Differentiation of services may be implemented by a traffic prioritization approach allowing priority traffic to receive a higher share of the link bandwidth.

1. The Traffic Prioritization Problem

Consider a directed network of N nodes with index set \mathcal{N} and L links with index set \mathcal{L} and a set of service classes \mathcal{S} . Let $r_{i,j}^s$ denote the class s traffic offered to node pair (i, j) . We wish to find an optimal link flow vector \mathbf{f}_{opt} such that

$$P(\mathbf{f}_{\text{opt}}) = \min_{\mathbf{f}} \sum_{s \in \mathcal{S}} \sum_{\ell \in \mathcal{L}} P_{\ell}(f_{\ell}^s)$$

subject to the constraints

$$\sum_{s \in \mathcal{S}} f_{\ell}^s \leq C_{\ell}$$

for all $\ell \in \mathcal{L}$ where f_{ℓ}^s is the class s flow on link ℓ , C_{ℓ} is the capacity of link ℓ , $\mathbf{f} = (f^1, f^2, \dots, f^S)$, $P_{\ell}(f_{\ell}^s)$ is a link penalty function and $f^s = (f_1^s, \dots, f_L^s)$.

2. Traffic prioritization Algorithms

Weighted Traffic Increment

In a multi-path environment where traffic prioritization provides different QoS levels to the offered traffic, differentiated services may be supported by partitioning the offered traffic $r_{i,e}$ into traffic classes $\{r_{i,e}^s\}$ which are allocated load factors $\{\phi_{i,e}^s\}$ expressing the importance accorded to the traffic classes $\{s \in \mathcal{S}\}$ and where priority traffic receives higher values of the load factor. This is done by reformulating the incremental load expression as:

$$a_{i,e} = m_{i,e} \sum_{s \in \mathcal{S}} \phi_{i,e}^s r_{i,e}^s \quad (43)$$

where $m_{i,e}$ is a multiplicative factor:

$$m_{i,e} = \begin{cases} m & : \text{ multiplicative} \\ \alpha & : \text{ additive} \end{cases}$$

$$m = \begin{cases} 1/2^k & : \text{ decrease} \\ 2^k & : \text{ increase} \end{cases}$$

and $\phi_{i,e}^s$ is a service factor:

$$\phi_{i,e}^s = \begin{cases} 1 & : \text{ mono-service} \\ \phi^s & : \text{ multi-service} \end{cases}$$

where $k \geq 0$ is an integer value, $s \in \mathcal{S}$ is the service class, ϕ^s is the load factor corresponding to the service class s and $\alpha \geq 0$ represents either a decrease operation ($0 \leq \alpha \leq 1$) or an increase operation ($1 \leq \alpha \leq 2$).

5.4 Fairness Models

Multi-service WTD: WTC algorithm

Service differentiation can be incorporated in the WTD algorithm by a weighted cost and weighted traffic algorithm (WCT) which allocates to each traffic requirement $r_{i,e}^s$ a load factor $\phi_{i,e}^s$ expressing the importance accorded by the network operator to traffic class s and assigning to a path $p \in \mathcal{A}_{i,e}$ the traffic $\delta_p(i, e)$ computed based on a cost factor w_p^β . The WCT algorithm executes the same steps as the WTD algorithm except for the flow increment $a_{i,e}$ and the traffic flow $\delta_p(i, e)$ which are computed as follows:

- **compute**($a_{i,e}$): $a_{i,e} = m_{i,e} \sum_{s \in \mathcal{S}} \phi_{i,e}^s r_{i,e}^s$
- **compute**($\delta_p(i, e)$): $\delta_p(i, e) = \frac{w_p^\beta}{\sum_{k \in \mathcal{P}_{i,e}} w_k^\beta} a_{i,e}$

where w_p^β is a path cost factor related to the load on path p , $r_{i,e}^s$ is the class s traffic offered to the source-destination (i, e) , $a_{i,e}$ is the aggregate traffic increment associated with all the traffic requirements $r_{i,e}^s$ and $\phi_{i,e}^s$ is a traffic factor related to the priority allocated to the traffic requirement $r_{i,e}^s$.

Multi-service FLD: FLD_s algorithm

Similarly, the FLD algorithm may be extended to a differentiated services algorithm FLD_s by executing the same steps as the FLD algorithm except for the traffic increment which will be computed by equation (43).

5.4 Fairness Models

Bandwidth sharing algorithms are greatly influenced by how transmission rates are allocated to the existing paths. Different adaptive rate models effecting different fairness levels in the network were evaluated in (Massoulié and J.Roberts, 2000). These include: maximum-throughput, max-min, minimum potential delay, proportional fairness and weighted shares fairness. A more general model based on a general bandwidth sharing criterion called α -bandwidth allocation was presented in (Mo and Walrand, 2000). These models differ as to how the traffic is allocated to the shortest and longest paths. This section summarizes the results presented in (Massoulié and J.Roberts, 2000) and addresses the problem of network fairness when deployed in load-balancing schemes implementing the FLD and WTD algorithms.

5.4.1 The Traffic Regulation Problem

Consider a directed network of N nodes with index set \mathcal{N} and L links with index set \mathcal{L} where each link $\ell \in \mathcal{L}$ has a capacity $C_\ell > 0$. We consider a set of flows competing for access to links on a route. Let \mathcal{R} denote the set of routes. We denote $\ell \in r$ when route r goes through link ℓ . Let f_r

denote the traffic flow allocated to route r . We wish to find an optimal flow vector f_{opt} such that

$$R(f_{opt}) = \max_f \sum_{r \in \mathcal{R}} \sum_{\ell \in r} R_\ell(f_r)$$

subject to the feasibility constraints

$$\sum_{r \in \mathcal{A}_\ell} f_r \leq C_\ell$$

for all $\ell \in \mathcal{L}$ where f_r is the rate on path r , \mathcal{A}_ℓ is the set of routes traversing link ℓ , $f = (f_1, f_2, \dots, f_L)$ and $R_\ell(f_r)$ is a link penalty function.

5.4.2 Mono-path Fairness Characteristics

Network fairness has long been considered as the guiding parameter for bandwidth sharing in the best effort Internet where there is no explicit admission control or quantitative service assurances. The classical network fairness model is the Max-Min fairness model informally defined by (Jaffe, 1981) as the allocation of resources such that each user's throughput is at least as large as that of any other user. Fairness models may be classified into two classes: macroscopic models where fairness is examined along a path and local fairness where fairness is examined on a per link basis.

The traffic regulation problem provides different fairness characteristics depending on the nature of the reward function used. This subsection presents fairness characteristics exhibited by a linear network model. In this model, the traffic offered to a long path interferes with the traffic requirements of L other single link paths. The single-link paths do not interfere among themselves. The flow allocation on the longer path is referred to as f_0 while the one-link path allocations are referred to as f_r . We consider a network where each network link has C units capacity.

Parking Lot Scenario

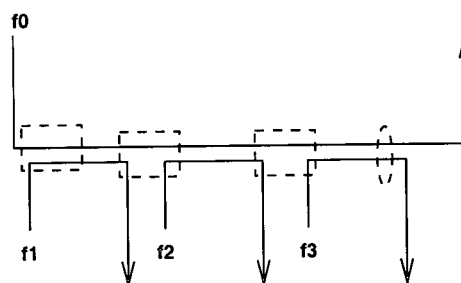


Figure 5.3: The “linear network” model

Different adaptive rate algorithms were presented in (Massoulié and J.Roberts, 2000). They are based on different objective functions implementing different fairness levels in the network. These reward functions include:

5.4 Fairness Models

- **Max Throughput reward function:**

$$R_\ell(f_r) = f_r.$$

Bandwidth sharing schemes implementing max throughput in a linear model provide a total throughput of $L - (L - C)f_0$ which is maximized by the value $f_0 = 0$. This informally expresses a bandwidth sharing model which is maximized when no bandwidth share is allocated to the traffic requirement using the longest path. This may be an issue in a network where the network efficiency requires that some traffic flows collapse in favour of other flows.

- **Max-Min reward function:**

$$R_\ell(f_r) = \min_{\mathcal{R}}\{f_r\}$$

In the case of the linear network model where $x_i = 1$, max-min fairness achieves a total throughput of $(L + 1)/2$ units of flow by allocating to all paths a rate equal to $1/2$ flow units. In this model, the throughput ratio between longer and short routes is 1.

- **Minimum Potential reward function:**

$$R_\ell(f_r) = 1/f_r$$

A detailed analysis of the bandwidth sharing implementing minimum potential delay in a linear model is presented in the appendix. The analysis shows that:

1. The total throughput produced by minimum potential delay $1 + L - \sqrt{L}$ is superior to the throughput $(L + 1)/2$ produced by the max-min model when $L \geq 1$.
2. While max-min shares the network resources fairly between routes, minimum potential delay penalizes the longer routes. This is expressed by:

$$\gamma_0 \leq \gamma_i$$

where γ_0 and γ_i are the rates received by the longest route r_0 and any other shorter route r_i respectively.

- **Proportional Fairness reward function:**

$$R_r(f_r) = \frac{1}{R} \sum_{r \in \mathcal{R}} \log f_r$$

where f_r is the traffic flow through route r . It is shown in the appendix that for the linear model under consideration, proportional fairness:

- achieves the throughput $\gamma_0 = 1/(L + 1)$ on the long route and $\gamma_i = L/(L + 1)$ on short routes.
- achieves a total throughput $L - (L - 1)/(L + 1)$

It can be observed that:

1. The proportional fairness throughput $L - ((L - 1)/(L + 1))$ is greater than the max-min throughput $(L + 1)/2$ for $L > 1$, and
 2. In proportional fairness, longer routes are more heavily penalized than in max-min fairness. This model is expressed by the ratio longest/shortest routes which is equal to $1/L$ in the case of proportional fairness and 1 for Max-min fairness.
- **Weighted shares reward function:** Applying weighted shares generalizes the two transmission rate models presented above by weighting the shares allocated to the traffic requirements. A weighted share allocation model provides the potential for differentiation of services in IP networks. This is implemented for example by allocating a greater increment to priority traffic. Both max-min and minimum potential delay may be generalized by introducing a factor ϕ_r for each traffic requirement transmitted on route r such that an increase in this weight leads to an increase in the throughput λ_r . A generalization of the max-min criteria and the minimum potential delay to a weighted shares is given by:

Minimum Potential delay:

$$\sum_{\mathcal{R}} \phi_r / f_r$$

over the capacity constraints, and

Max-min fairness:

$$\min_{\mathcal{R}} (\phi_r f_r)$$

over the capacity constraints.

- **α -Bandwidth allocation reward function:** A general bandwidth sharing criterion called α -bandwidth was introduced by Mo and Walrand (Mo and Walrand, 2000). This general criterion is based on the reward function:

$$R_r(f_r) = \sum_{r \in \mathcal{R}} \frac{f_r^{1-\alpha}}{1-\alpha}$$

Where $\alpha \neq 1$ is a positive constant.

It can be observed that when $\alpha \rightarrow 2$, α -bandwidth leads to a max-min criterion and when $\alpha \rightarrow 1$ this criterion leads to the potential minimum delay. A generalization of the α -bandwidth sharing to the weighted shares model is given by:

$$R_r(f_r) = \sum_{r \in \mathcal{R}} w_r x_r^\alpha \frac{f_r^{1-\alpha}}{1-\alpha}$$

5.4.3 Multi-path Fairness Characteristics

WTD fairness

The traffic allocation model used has an impact on the solution computed by the traffic distribution algorithm and the performance level reached. The idea behind the weighted traffic distribution

5.4 Fairness Models

algorithm (*WTD*) based on a power β of the path costs is to provide a more general traffic distribution criteria that shares the traffic between shorter and longer paths differently according to the value of the power. It is commonly admitted that a fair allocation of bandwidth allocates more traffic to the shortest paths to approach a proportional fair solution (as defined by the TCP protocol fairness). A second advantage of the use of the power of path costs in a traffic distribution scheme aiming at sharing a given amount of traffic among several pre-computed paths is to effect a filling procedure where better (shortest) paths are filled first and reserve capacity on longer routes used in further traffic allocations.

WTD may compute different solutions depending on the value of the power used. These solutions may differ from those computed by *FLD*. An analysis of the *WTD* traffic distribution using the “eye network” presented in chapter 2 showed that weighted traffic distribution may compute different solutions. These different solutions mimic different rate allocation schemes depending on the value of the power.

It was shown in chapter 2 that:

- for $(\beta \rightarrow \infty)$, *WTD* implements a shortest path routing policy where longer routes are heavily penalized compared to shorter routes.
- for $(\beta \rightarrow 1)$, *WTD* implements a proportional sharing policy where longer routes are less penalized compared to shorter routes.
- for $(\beta \rightarrow 0)$, *WTD* implements an equal share policy mimicking the max-min fair allocation where longer and short routes receive the same throughput.

FLD fairness

It was observed in chapter 2 that different solutions (expressed by the number of active paths identified by the algorithm and the distribution of the traffic to paths) exist to the problem of flow allocation when deployed in the “fish” network presented in Figure 2.5.

The fairness of mono-path routing schemes refers to how different traffic flows share interfering links. In multi-path routing schemes, the network fairness may refer to how the traffic offered to a source-destination pair is shared among the paths identified on that source-destination pair. It is expected that a fairer allocation should distribute the traffic offered to as many available paths as possible to share the load between these paths for improved performance (delay, throughput). However, operational requirements may prefer a traffic distribution based on a minimum number of paths.

A vector representation of the fairness achieved by the *FLD* algorithm in the “fish” network is presented in Figure 5.4. In this figure, the z axis represents the traffic carried by path $r_{(1,3,4,6)}$ and the y axis represents the throughput received by the other paths. The lines $y = a - z$ and

$y = b - z$ represent optimal traffic allocations achieved by the FLD algorithm on paths $r_{(1,3,5,6)}$ and $r_{(2,3,4,6)}$ respectively where z is the traffic carried by path $r_{(1,3,4,6)}$ and $a = 0.5$ and $b = 1.0$ are the traffic flows offered to the O-D pairs (1,6) and (2,6) respectively. These lines are called “efficiency” lines.

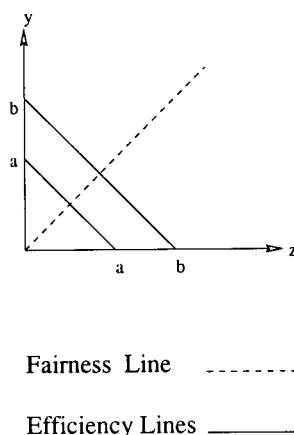


Figure 5.4: the “multipath fairness” model

The allocations for which $z = \frac{a}{2}$ and $z = \frac{b}{2}$ are the two points guaranteeing multipath fairness for the traffic offered to the O-D pairs (1,6) and (2,6) respectively. These points belong to a “fairness” line represented by the equation $y = z$. It can be observed that the two fairness values $a/2$ and $b/2$ produce two different sets of paths in the “fish” network: (1) four paths $r_{(1,3,4,6)}$, $r_{(1,3,5,6)}$, $r_{(2,3,4,6)}$ and $r_{(2,3,5,6)}$ computed by the FLD algorithm for the fairness point $z = \frac{a}{2}$ and (2) three paths $r_{(1,3,4,6)}$, $r_{(2,3,4,6)}$ and $r_{(2,3,5,6)}$ computed by the FLD algorithm for the fairness point $z = \frac{b}{2}$.

The fairness point $z = \frac{b}{2}$ discovers two active paths for the traffic offered to the O-D pair (2,6) and only one path for the traffic offered to the O-D pair (1,6). This shows that in multi-path routing, achieving multipath fairness for some traffic flows may result in unfairness for other flows.

While achieving multipath fairness for the two traffic flows by sharing each of these traffic flows among the two existing paths, the FLD algorithm finds more paths for the fairness point $z = \frac{a}{2}$. This example shows that, besides the computation of solutions providing multi-path fairness, FLD schemes raise the problem of tradeoff between fairness and engineering performance.

5.5 Simulation Analysis

We conducted simulation experiments using Network Simulator (McCanne and Floyd) to evaluate how different UDP flows are shared among label switched paths in a routing environment implementing the routing architecture of chapter 4. The relevant performance parameters are the throughput received by UDP flows, the traffic loss, the ratio throughput/signalling and the

5.5 Simulation Analysis

network fairness expressing how the traffic is shared among competing traffic flows. The signalling is expressed by the number of routing updates computed to adjust the transmission rate of the traffic upon threshold loading.

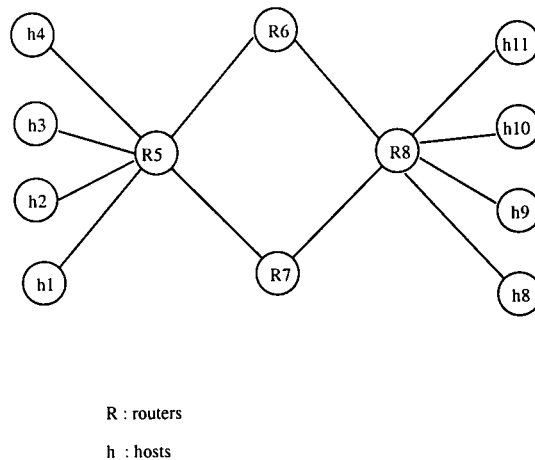


Figure 5.5: the “test network” model

The network model studied is presented in Figure 5.5. We consider four traffic flows f_1 , f_2 , f_3 and f_4 originating from the four source hosts h_1 , h_2 , h_3 and h_4 to the destination hosts h_8 , h_9 , h_{10} and h_{11} respectively. These flows are competing for bandwidth on the two available label switched paths $r_{(5,6,8)}$ and $r_{(5,7,8)}$ to transmit traffic to the destination hosts h_8 , h_9 , h_{10} and h_{11} . Periodically, the traffic flows are increased using traffic increments which are halved when the two label switched paths are above a threshold expressed by a link utilization above 0.8.

A first set of simulation experiments was conducted using an explicit routing scheme where the flows f_1 and f_2 are routed over the first label switched path $r_{(5,6,8)}$ while the flows f_3 and f_4 are routed over the second label switched path $r_{(5,7,8)}$. The traffic increments considered are $a_1 = a_2 = 11$ and $a_3 = a_4 = 7$ where a_1 , a_2 , a_3 and a_4 are related to the flows f_1 , f_2 , f_3 and f_4 respectively.

The results are presented in Tables 5.1 and 5.2. Table 5.1 refers to a network model where all the links have a capacity $C_\ell = 40$ units while the results presented in Table 5.2 refer to a model where the bandwidth of links (5, 6) and (6, 8) are equal 60 units.

The results show that the explicit routing model achieves max-flow routing where some flows may receive a very low throughput on their explicit route. This is illustrated by the two flows f_2 and f_3 which received very little throughput compared to flows f_1 and f_4 according to the results presented in Table 5.1. The results presented in Table 5.2 show an increase in the throughput of flows f_2 and flow f_3 resulting from the increase of the bandwidth of the label switched path $r_{(5,6,8)}$. This suggests that the problem of unfairness resulting from the max-flow routing implemented by an explicit routing model can be solved by over-provisioning the label switched path.

We conducted a second set of simulation experiments to evaluate how the WTD routing model compares to the explicit routing model. The results are presented in Tables 5.3 and 5.4. The traffic increments used to compute the results presented in Table 5.3 are $a_1 = a_2 = 11$ and $a_3 = a_4 = 7$ and the power $\beta = 1$ in a network where all links have capacity $C_\ell = 40$ units. The results presented in Table 5.3 refer to a feedback factor $k = 1/2$ consisting in halving the traffic increment upon congestion while the results presented in Table 5.4 refer to a feedback factor $k = 3/4$ consisting in multiplying the traffic increments by $3/4$ upon congestion. These results reveal that:

- WTD achieves a max-min fair allocation of the traffic by distributing the traffic equally between the available paths. This is illustrated by the equal flows f_1 and f_2 sent to hosts h_8 and h_9 and the equal flows sent to hosts h_{10} and h_{11} .
- Halving the traffic upon congestion results in less loss compared to the use of a feedback factor $k = 3/4$. Though applied in a routing environment using UDP flows, this compares well the congestion model used in the TCP Reno model consisting in halving the TCP window size upon congestion.
- The highest increment flow receives a higher throughput compared to the lowest increment flow. This validates the differentiated services model based on a traffic prioritization by sharing the LSP bandwidth differently.

We conducted a third set of simulation experiments to evaluate the fairness achieved by the WTD approach under different values of the power. The results are presented in Table 5.5. These results confirm that WTD may achieve different fairness characteristics depending on the value of the power β .

Flow	throughput	loss
f_1	41,447	23,516
f_2	49	2,490
f_3	4,264	628
f_4	37,231	7,817
Total	82,992	34,451

Table 5.1: Explicit routing.

Flow	throughput	loss
f_1	41,447	23,516
f_2	20,780	4,541
f_3	38,566	8,807
f_4	2,930	8,091
Total	103,741	44,955

Table 5.2: Explicit routing.

Flow	throughput	loss
f_1	16,508	22,740
f_2	16,508	22,740
f_3	24,988	6,079
f_4	24,988	6,079
Total	82,992	57,638

Table 5.3: WTD using traffic halving.

Flow	throughput	loss
f_1	11,683	35,370
f_2	11,683	35,370
f_3	29,813	7,249
f_4	29,813	7,249
Total	82,992	85,238

Table 5.4: WTD using a feedback factor 3/4.

Flow	Power=1		Power=10		Power=25		Power=55	
	flow	loss	flow	loss	flow	loss	flow	loss
f_1	38,996	20,806	31,007	32,469	41,496	39,646	41,496	39,616
f_2	7,826	24,637	22,648	26,654	19,260	4,682	1,709	1,394
f_3	33,669	8,190	18,848	4,582	22,236	24,245	39,787	12,209
f_4	23,248	2,659	31,236	3,676	14,500	0	14,523	0
Total	103,739	56,292	103,739	67,381	97,492	68,573	97,515	53,219

Table 5.5: WTD using different power values.

	DropTail	RED	SFQ
Total throughput	82,992	103,740	103,740
Total Loss	57,638	42,193	42,192
Throughput/Signalling ratio	11,856	14,820	14,820

Table 5.6: WTD using different queueing models.

We compared through simulation the throughput received by different UDP flows under different queueing models. The queueing models considered are FIFO (First-In-First Out), RED (random Early Detection) and SFQ (Stochastic Fair Queueing). The results are presented in Table 5.6 and Figures 5.6, 5.7 and 5.8.

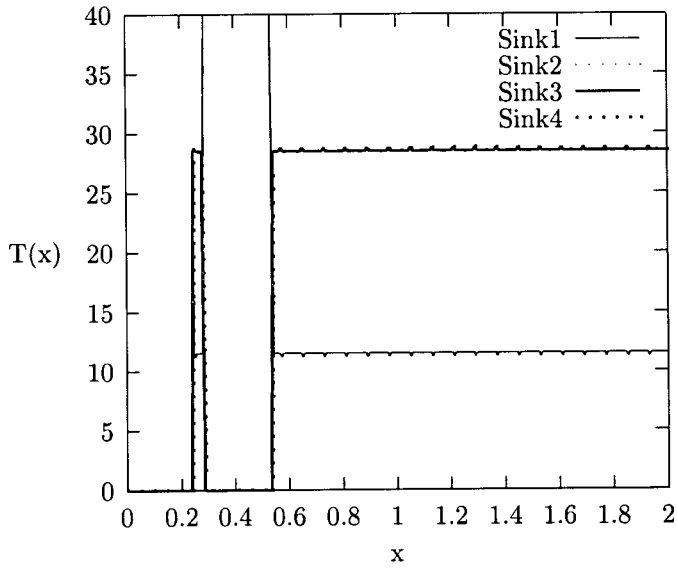
Figure 5.6 refers to the FIFO queueing based on the Drop-Tail model while Figure 5.7 refers to the RED queueing model and Figure 5.8 refer to the SFQ queueing model.

These figures display the shape of the throughput received by different UDP flows and the packet loss during the simulation period.

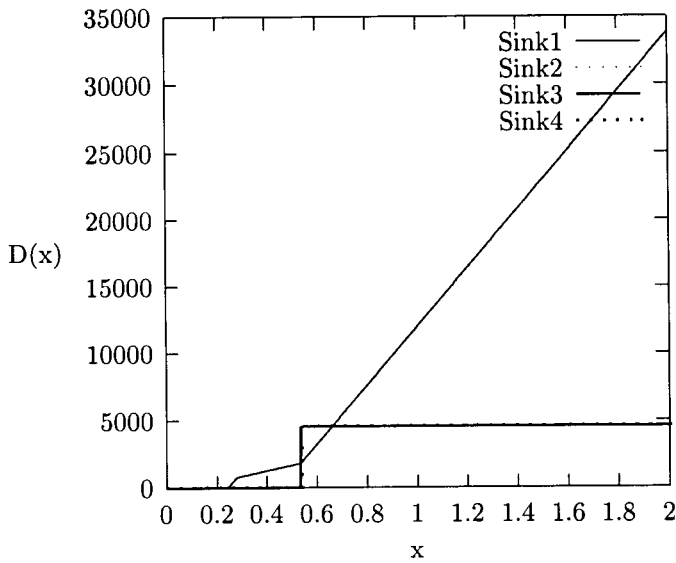
The results in Table 5.6 reveal that the overall throughput received by UDP flows and the throughput/signalling ratio is improved under the RED and SFQ queueing models. The shape of the throughput displayed by these figures shows that the SFQ queueing model effects an equal distribution of the traffic to the different UDP flows compared to the DropTail and RED queueing models.

5.6 Conclusion

This chapter validates the routing architecture proposed in chapter 4 by comparing the throughput received by UDP flows in a simulated MPLS environment implementing the WTD load balancing approach with an explicit routing scheme implementing a max-flow routing approach. The performance expressed in terms of throughput, packet loss and network fairness of the routing architecture have been evaluated using different WTD parameters and queueing models. Simulation reveals that routing the traffic based on the architecture proposed in chapter 4 results in improved network efficiency and flexibility providing the network manager with an effective control over the network infrastructure and an easy implementation allowing the deployment of different network fairness models.

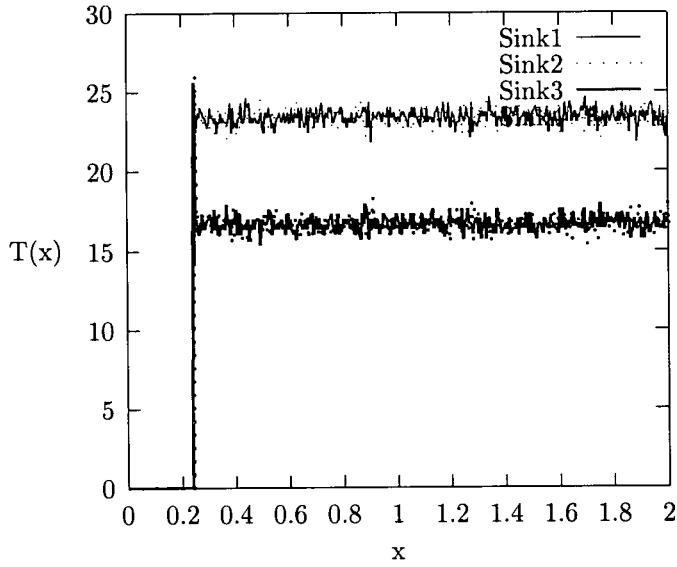


Throughput: DropTail queueing model

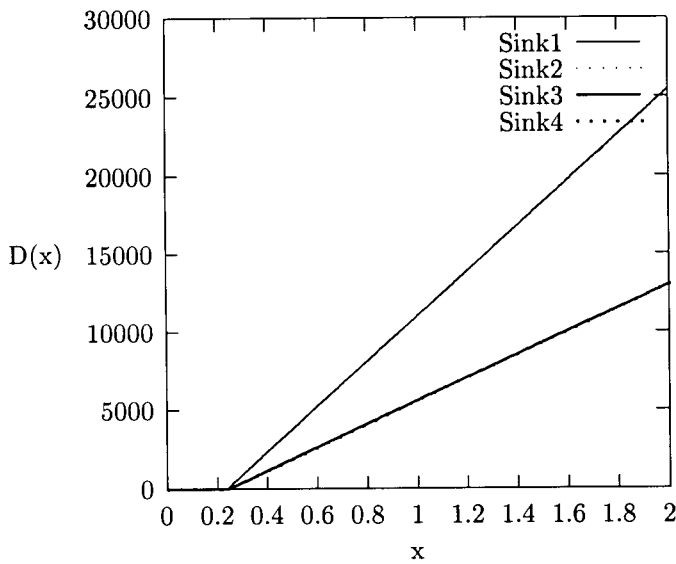


Packet loss: DropTail queueing model

Figure 5.6: DropTail queueing model

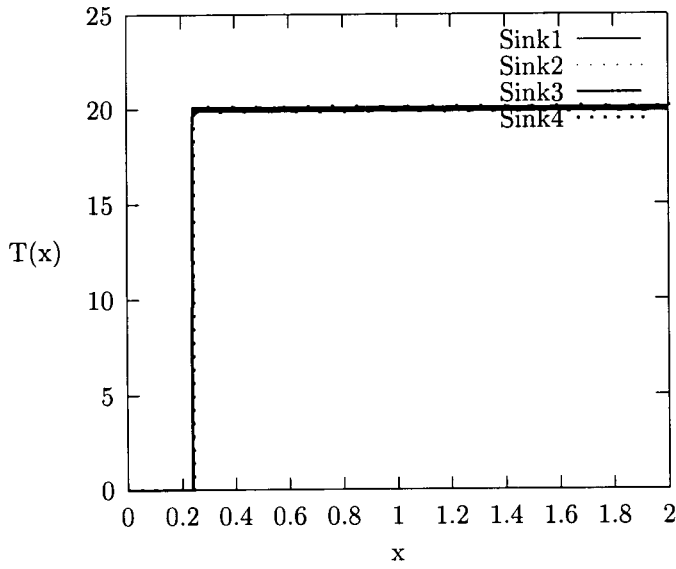


Throughput: RED queuing model

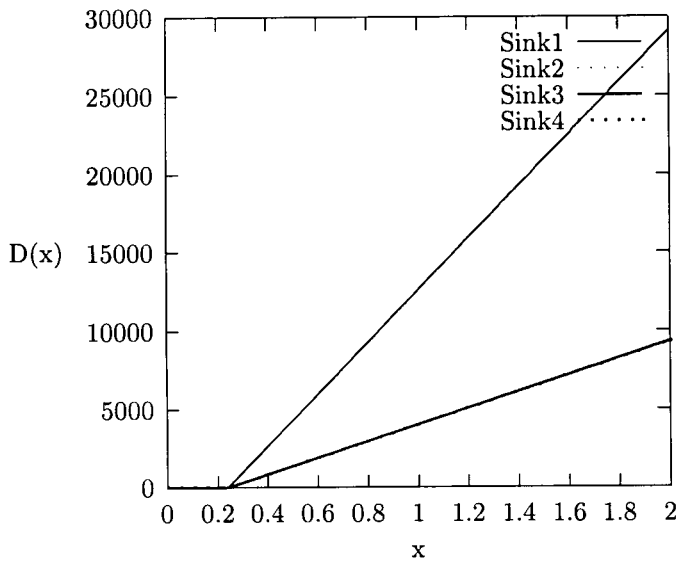


Packet loss: RED queuing model

Figure 5.7: RED queuing model



Throughput: SFQ queuing model



Packet loss: SFQ queuing model

Figure 5.8: SFQ queuing model

Chapter 6

Conclusion

6.1 Thesis Summary

The focus on this thesis is on mechanisms to be deployed in MPLS networks for traffic engineering label switched paths (LSPs). Two main topics were addressed in this thesis.

First, we present flow optimization models implementing a pre-planned scheme separating path selection from bandwidth allocation to compute optimal LSPs. These models are presented in chapter 2 and chapter 3. The basic model presented in chapter 2 is complemented by the threshold routing approach presented in chapter 2 to avoid the bandwidth fragmentation which may result from the opportunistic routing approach implemented by the basic model. These models allocate the traffic in small increments to avoid the computational complexity of finding feasible paths. These models are evaluated using a variant of the well-known “flow deviation method” and another algorithm which distributes the traffic among the available paths based on weights assigned to these paths. Using these models, we conducted experiments which showed that pre-planned flow allocation is an approach which finds near optimal label switched paths in a computationally effective way.

Second, we propose a routing architecture which implements a traffic regulation approach based on a network cooperation between the end applications, the ingress and the egress of the MPLS network. The proposed routing architecture is presented in chapter 4. The traffic regulation model is presented in chapter 5 and validated through simulation using the NS software package. The allocation of the traffic in small fractions of the offered traffic to mimic congestion avoidance mechanisms deployed for the TCP protocol and the distribution of the traffic among available paths based on the weighted distribution allocation are the basic concepts adopted in the traffic regulation model. Simulation revealed that routing the traffic based on our proposed architecture results in improved performance and network flexibility compared to the deployment of non load-balancing explicit routing schemes.

6.2 Future Work

The work contained in this thesis may be extended and complemented by:

- The results presented in chapter 5 refer to a traffic regulation model implementing the *WTD* approach to compute a global optimum using local information recorded in the *LSB* database. This model may be extended to an integrated approach implementing the *WTD* approach periodically to effect congestion avoidance, the *FLD_d* approach to move part of the traffic of a over-loaded path to under-loaded paths and the *FLD_i* to move part of the traffic of over-loaded paths towards an under-loaded or new activated path.
- Though the allocation of the traffic in small increments of the offered traffic has being proposed and evaluated through simulation, effective methods to compute the traffic increment constitute an important aspect of the traffic regulation model which is yet to be investigated. The market-pricing paradigm can provide more insights on how the increments can be efficiently allocated to the different traffic flows competing for bandwidth on network paths. Future research work will focus on this aspect.
- Though being successfully validated on a long-term time scale in a simulation environment, the flow allocation models need to be extended to include local reconfiguration mechanisms deployed on a short-term and intermediate time scale. This topic has been reserved for future research work.

Appendix

.1 Minimum Potential Delay

Minimum Potential Delay implements a bandwidth sharing model where the route reward function has the form $R_\ell(f_r) = 1/f_r$. Let γ_0 and γ_i denote the throughput received respectively by the longer routes and shorter routes. The minimum potential delay model will realize a rate allocation corresponding to the following equation in the linear model where each link has one unit capacity:

$$x_0\gamma_0 + x_i\gamma_i = 1$$

where γ_0 and γ_i are the number of longer routes that share link i and the number of shorter routes sharing link i respectively. This equation is rewritten as:

$$\gamma_i = \frac{1 - x_0\gamma_0}{x_i}$$

The minimum potential delay formulation thus becomes:

$$R(\gamma_0) = \frac{x_0}{\gamma_0} + \sum_{i=1}^L \frac{x_i^2}{(1 - x_0\gamma_0)}$$

where L is the number of one-link routes which share the traffic offered to the linear network between the longest route and a shorter route. This expression is minimized for values which zero its derivative with respect to the throughput γ_0 . This gives a value of γ_0 expressed by:

$$\gamma_0 = \frac{1}{x_0 + \sqrt{\sum_{i=1}^L x_i^2}}$$

If $x_0 = 1$ and $x_i = 1$, the throughput of longer and shorter routes are given by:

$$\gamma_0 = \frac{1}{1 + \sqrt{L}}$$

$$\gamma_i = \frac{\sqrt{L}}{1 + \sqrt{L}}$$

Replacing these values in the expression of the total throughput $x_0\gamma_0 + \sum_{i=1}^L x_i\gamma_i$ gives a total throughput for the linear network model of $1 + L - \sqrt{L}$.

It can be observed that:

1. This throughput is superior to the throughput offered by the max-min rate allocation model which is given by:

$$\frac{L+1}{2} \leq 1 + L - \sqrt{L}$$

2. While Max-min shares the network resources fairly between routes, Minimum potential delay penalizes longer routes. This is expressed by:

$$\gamma_0 \leq \gamma_i$$

A more elaborate Minimum Potential Delay penalty function is given by

$$T = \frac{1}{R} \sum_{p \in \mathcal{P}} \frac{B_p}{f_p} - T_p \log f_p$$

where B_p is the window size of the TCP traffic flowing through route p and T_p is the round-trip experienced by the traffic on path p .

.2 Max-Min Fairness

Max-Min fairness implements a bandwidth sharing model where the route reward function is given by:

$$R_\ell(f_r) = \min_{\mathcal{R}} \{f_r\}$$

Three key features are involved in Max-Min fairness:

- resources are allocated in increasing order of demand,
- a user is never allocated a share higher than its demand, and
- all users with unsatisfied demands are allocated equal shares.

In the linear network implementing max-min fairness, transmission rates are allocated as follow:

$$f_r = \begin{cases} \frac{1}{x_0 + \max_{\ell \geq 1} x_\ell} & : \text{ for } r \in \mathcal{R}_i \\ \frac{1}{x_\ell} \left(1 - \frac{x_0}{x_0 + \max_{\ell \geq 1} x_\ell}\right) & : \text{ if } r \in \mathcal{R}_\ell, \ell \geq 1, x_i \geq 0 \end{cases}$$

It can be observed that in a linear model implementing Max-min fairness, a fair share of bandwidth is allocated to longer as well as shorter routes and the rates on longer and shorter routes are given by:

$$\gamma_0 = 0.5$$

$$\gamma_i = 0.5$$

and

$$x_0 \gamma_0 + \sum_{i=1}^L x_i \gamma_i = \frac{(L+1)}{2}$$

where $x_0 = 1$ is the number of longer routes that share link i and $x_i = 1$ is the number of shorter routes sharing link i .

If the $x_i = 1$, Max-min fairness realizes a total throughput of $(L + 1)/2$ by allocating to all paths a rate equal to $1/2$. In this model, the throughput ratio between longer and shorter routes is 1.

.3 Proportional Fairness

Proportional fairness implements a bandwidth sharing model where the route reward function is given by:

$$R(f_r) = \log f_r$$

Where f_r is the traffic flowing through route r . The expression $x_0\gamma_0 + x_i\gamma_i = 1$ holds again for a linear network where $1 \leq i \leq L$. The optimal bandwidth sharing corresponds to a value of γ_0 which maximizes the function:

$$R(\gamma_0) = x_0 \log(\gamma_0) + \sum_{i=1}^L x_i \log\left(\frac{1 - x_0\gamma_0}{x_i}\right)$$

This optimum is given by values of γ_0 which zero the derivative of the function $R(\gamma_0)$ with respect to γ_0 . This is expressed by the relation

$$\frac{x_0}{\gamma_0} = \sum_{i=1}^L \frac{x_i x_0}{1 - x_0\gamma_0}$$

which gives

$$\gamma_0 = \frac{1}{x_0 + \sum_{i=1}^L x_i}$$

for $x_i = 1$ and $0 \leq i \leq L$,

$$\gamma_0 = \frac{1}{L + 1}$$

and

$$\gamma_i = \frac{L}{L + 1}.$$

These equations give a total throughput of $L - (L - 1)/(L + 1)$.

Bibliography

- G. Ash. Traffic engineering and QoS methods for IP-, ATM- and TDM-based multiservice networks. In *INTERNET DRAFT draft-ietf-qos-routing-01.txt*, March 2001.
- D. Awduche, J. Malcolm, M. O'Dell, and J. McManus. Requirements for traffic engineering over MPLS. In *INTERNET DRAFT draft-awduche-mpls-traffic-eng-00.txt*, April 1998.
- A. Bagula and A.E. Krzesinski. Traffic engineering label switched paths in IP networks using a pre-planned flow optimization model. In *Proceedings of the 9th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pages 70–77, August 2001.
- F. Barahona. Network design using cut inequalities. *SIAM Journal on Optimization*, 6:823–837, 1996.
- L. Benmohamed, S. Dravida, P. Harshavardhana, A. Lau, and A. Mittal. Designing IP networks with performance guarantees. *Bell Labs Technical Journal*, 3(4):273–296, 1998.
- J.E. Burns, T.J. Ott, J.M. De Kock, and A.E. Krzesinski. Path selection and bandwidth allocation in MPLS networks: a non-linear programming approach. In *Proceedings of The International Society of Optical Engineering*, pages 15–26, August 2001.
- A. Charny, D. D. Clark, and R. Jain. Congestion control with explicit rate indication. In *Proc. ICC 95*, pages 1954–1963, June 1995. URL citeseer.nj.nec.com/charny95congestion.html.
- J. Chifflet, P. Mahey, and V. Reynier. Proximal decomposition for multi-commodity flow problems with convex cost. *Telecommunication Systems*, 3:1–10, 1994.
- D.D. Clark. Adding service discrimination to the internet. *Telecommunications Policy*, 20:169–181, 1996.
- B. Davie, P. Doolan, and Y. Rekhter. *Switching in IP Networks: IP Switching, Tag Switching and Related Technologies*. Morgan Kaufman Publishers Inc., 1998.
- Diffserv. Differentiated Services Working Group (Diffserv). IETF.
- B. Fortz and M. Thorup. Internet traffic engineering by optimizing OSPF weights. In *IEEE INFOCOM 2000*, pages 519–528, 2000.

- L. Fratta, M. Gerla, and L. Kleinrock. The flow deviation method: an approach to store-and-forward communication network design. *Networks*, 3:97–133, 1973.
- P. Gevros, J. Crowcroft, P. Kirstein, and S. Bhatti. Congestion control mechanisms and the best effort service model. In *IEEE Network Special Issue on Control of Best Effort Traffic*, 2001.
- Intserv. Integrated Services Working Group (Intserv). IETF.
- J.M. Jaffe. Bottleneck flow control. *IEEE Trans. on Comm.*, COM-29(7):954–962, 1981.
- R. Jain and D. Chiu. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems*, 17:1–14, 1989.
- B. Jamoussi. Constraint-based LSP setup using LDP. In *INTERNET DRAFT draft-ietf-mpls-cr-ldp-03.txt*, September 1999.
- F.P. Kelly, A. Maulloo, and D. Tan. Rate control for communication networks: shadow prices,proportional fairness and stability. *Journal of the Operational Research Society*, 49, 1998.
- A. Kershenbaum. *Telecommunication Design Algorithms*. McGraw-Hill, New York, NY, 1993.
- A. Kershenbaum, P. Kermani, and G. Grover. MENTOR: An algorithm for mesh network topological optimization and routing. *IEEE Trans. on Comm.*, 39(4):503–513, 1991.
- M. Kodialam and T.V. Lakshman. Minimum interference routing with applications to MPLS traffic engineering. In *IEEE INFOCOM 2000*, pages 884–893, 2000.
- W. Lai. Capacity engineering of IP-based networks with MPLS. In *INTERNET DRAFT draft-wlai-tewg-cap-eng-00.txt*, March 2000.
- J.K. Mackie-Mason and H.R. Varian. Pricing the congestible network resources. *IEEE J. Selected Areas Comm.*, 13:1141–1149, 1995.
- L. Massoulie and J.Roberts. Bandwidth sharing and admission control for elastic traffic. *Telecommunication Systems*, 15:185–201, 2000.
- S. McCanne and S. Floyd. The LBNL network simulator. In *URL <http://www-nrg.ee.lbl.gov/ns/>*, Lawrence Berkeley Laboratory.
- J. Mo and J. Walrand. Fair end-to-end window-based congestion control. *IEEE/ACM Trans. on Networking*, 8-5:556–567, 2000.
- A.M. Odlyzko. Paris metro pricing for the internet. In *Proc.ACM Conference on Electronic Commerce(EC'99)*, pages 140–147, 1999.
- OSI. Open Systems Interconnection. In *International Organization for Standardization. Ref. no. ISO8073-1986 (E)*, 1986.
- S. Plotkin. Competitive routing of virtual circuits in ATM networks. *IEEE J. Select Areas in Comm.*, 13(1):1128–1136, August 1995.

- J. Sairamesh. *Economic Paradigms for Information Systems and Networks*. PhD thesis, Columbia University, New York, NY, 1997.
- J. Sairamesh, D. Ferguson, and T. Yemini. An approach to pricing, optimal allocation and quality of service provisioning in high speed packet networks. In *Proceedings of the INFOCOM95*, 1995.
- D. Thaler and C. Hopps. Multipath issues in unicast and multicast next-hop selection. In *INTERNET DRAFT draft-thaler-multipath-04.txt*, June 1999.
- C. Villamizar. MPLS optimized multipath (MPLS-OMP). In *INTERNET DRAFT draft-ietf-mpls-omp-00.txt*, August 1999.
- Y. Wang and Z. Wang. Explicit routing algorithms for internet traffic engineering. In *IEEE Proc. ICCCN*, pages 582–588, 1999.
- A.E.I. Widjaja. MATE: MPLS adaptive traffic engineering. In *INTERNET DRAFT draft-widjaja-mpls-mate-00.txt*, August 1998.