School of Electronic Engineering

Dublin City University

Thesis directed by Dr Martin Collier

Traffic Control in Packet-Switched Networks

by Ljiljana Adamovic

A thesis submitted for the degree of Doctor of Philosophy

September 20, 2010

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy *is entirely my own work, that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.*

Signed:
ID No.: 52143309
Date: 20/09/2010

To Daire, Mark, and Lara

Acknowledgement

I would like to thank my supervisor, Dr Martin Collier, for giving me the opportunity to do research in the area of routing in packet-switched networks, and for his advices and help during my research years in DCU.

I am deeply grateful to Dr Claire Bohan, Prof. Gary Murphy, and Dr Noel Murphy of DCU, Dublin, to the examiners of my PhD thesis, Dr Gabriel-Miro Muntean of DCU, Dublin, and Prof. Laurie Cuthbert, of QMUL, London, and to my husband, Daire Quinlan. Without their help this thesis would not have been completed.

Many thanks also to Dr John Schormanns of QMUL, London, for his comments on my work.

Abstract

This thesis examines traffic control options available in two existing routing solutions in packet-switched networks. The first solution is the shortest path hop-by-hop routing deployed with the OSPF or IS-IS routing protocol and the IP forwarding protocol. This is the initially deployed and still the most popular routing solution in the Internet. The second solution is explicit routing implemented with the RSVP-TE or CR-LDP signalling protocol and the MPLS forwarding protocol. This is the latest solution to have become widely deployed in the Internet. The thesis analyses the limitations of the two routing solutions as tools for traffic control and yields new insights that can guide the analysis and design of protocols involved in the process. A set of recommendations for modifications of the existing protocols is provided which would allow for a range of new traffic control approaches to be deployed in packet-switched networks.

For future routing solutions which comply with the proposed recommendations two new algorithms are presented in the thesis. They are called the Link Mask Topology (LMT) algorithm, and the Link Cost Topology (LCT) algorithm. The two algorithms define a set of routing topologies and assign network traffic to routes available in these topologies aiming to simultaneously achieve high network throughput and fair resource allocation. While there are similarities in the operation of the two algorithms, their applicability is different as they allocate resources to multiple paths between two network nodes which are available in the defined routing topologies according to a different rule set. The LMT algorithm directs traffic sent between any pair of network nodes to a single route. The LCT algorithm directs traffic sent between a pair of network nodes to a number of routes. The performance of the two proposed algorithms is evaluated in the thesis with calculations comparing them to the shortest path routing algorithm in a number of test cases. The test results demonstrate the potentials of the two proposed algorithms in improving the performance of networks which employ shortest path routing.

List of Publications

A New Traffic Engineering Approach for IP Networks, by Ljiljana Adamovic and Martin Collier, in proceedings of CSNDSP 2004, Newcastle upon Tyne, 20-22 July 2004.

Internet Traffic Control with Multiple Topology Routing, by Ljiljana Adamovic, Karol Kowalik and Martin Collier, in proceedings of CSN 2005, Reunion Island, 17-22 April 2005.

List of Terms

routing solution: the set of tasks which have to be performed so that traffic in a network can reach its destination

forwarding topology: a topology formed by a set of unidirectional network edges, and a set of network nodes, along which packets are forwarded through the network

leaf node: a node in a forwarding topology with no incoming edges

root node: a node in a forwarding topology with no outgoing edges

directed line: the forwarding topology with a single leaf node and a single root node

directed tree: the forwarding topology with multiple leaf nodes and a single root node

directed ring: the forwarding topology without the root node and with no leaf nodes

forwarding table entry: a record for a particular node of the only outgoing edge that node has in a particular forwarding topology

forwarding table: the set of forwarding table entries for a single node

routing assignment: an assignment to a forwarding topology, for a particular node in that topology, of an address(es) reachable from another node in that topology

relevant routing assignment for a node: any routing assignment made at other nodes in the same network which causes packets to traverse that node.

entry node: a node in a forwarding topology with a routing assignment defined for that topology

exit node: a node in a forwarding topology with a reachable address(es) which appears in a routing assignment at another node in the same topology

routing table: the set of routing assignments for a single node

routing granularity: the factor which defines what is mapped to a forwarding topology in a single routing assignment, whether it is a single destination address, a set of destination addresses, or a single and destination address

coarse routing granularity: an indicator that a set of destination addresses is mapped to a forwarding topology in a single routing assignment

base routing granularity: an indicator that a single destination address is mapped to a forwarding topology in a single routing assignment

fine routing granularity: an indicator that a single source and destination address is mapped to a forwarding topology in a single routing assignment

routing algorithm: a set of rules for assigning destination addresses reachable from nodes in a forwarding topology to that forwarding topology

forwarding layer: a forwarding topology and the set of entry and exit nodes defined in that topology by all the routing assignments made for that forwarding topology

point-to-point forwarding layer: a forwarding layer with a single entry node and a single exit node

multipoint-to-point forwarding layer: a forwarding layer with multiple entry nodes and a single exit node

point-to-multipoint forwarding layer: a forwarding layer with a single entry node and multiple exit nodes

multipoint-to-multipoint forwarding layer: a forwarding layer with multiple entry nodes and multiple exit nodes

routing topology: an aggregate of a set of forwarding topologies

splitting algorithm for a routing topology: the algorithm that splits the routing topology into a set of forwarding topologies

routing state: a state of network and network traffic for which a single set of routing assignments is defined

routing state response algorithm: the algorithm that specifies the destination addresses reachable from the nodes in each forwarding topology in the current routing state

routing frequency: the frequency of updates of routing assignments in a running network

forwarding protocol: the protocol that forwards packets through the network by following forwarding instructions

routing protocol: the protocol that monitors routing state changes and provides forwarding instructions for the forwarding protocol in the current routing state.

forwarding instruction: an instruction for a single node indicating the outgoing edge of a node in a particular forwarding topology where a packet which carries particular information in its forwarding header needs to be forwarded

control protocol: the protocol that distributes configuration information from the network control centre to all network nodes and gathers routing state data for the network control centre

routing topology algorithms: algorithms which determine routing topologies for a single routing state in a routing solution for packet-switched networks

hop-by-hop IP routing: routing solution for packet-switched networks realized with the OSPF or IS-IS routing protocol and the IP forwarding protocol

explicit MPLS routing: routing solution for packet-switched networks realized with the RSVP-TE or CR-LDP signalling protocol and the MPLS forwarding protocol

label: an identifier

List of Figures

List of Tables

Contents

Chapter 1

Introduction

Packet-switched networks transfer information for network users between specified destinations. The largest existing packet-switched network, the Internet, is becoming the main means of communication on the planet within a short time of its establishment. The most significant function performed by packet-switched networks is routing of traffic to its destination. The most challenging task within the routing function is traffic control.

The constant increase in demands in packet-switched networks, and in particular the increase in demands for a high quality service, have instantiated extensive research for novel, sophisticated traffic control techniques which would ensure efficient, reliable, and fair network operation. There is a high demand for additional traffic control options in the existing routing solutions which would enable efficient traffic engineering [1] and Quality of Service (QoS) routing [2, 3, 4, 5]. The majority of potential obstacles in traffic control can be avoided by ensuring that, given the current demands for the network service, there are sufficient network capacities. However, this is not necessarily the most economical solution, and it is not always possible. The network capacities must be increased ahead of any potential increase in demands. As the accuracy of predictions regarding the growth in demands cannot be guaranteed, a severe over-provisioning is usually the safest bet for ensuring high quality network performance. The ongoing research for alternative options has lead to many new traffic control features in the existing routing solutions, but there are still many outstanding problems whose solutions remain to be found.

1.1 Motivation

The primary objective in controlling traffic in packet-switched networks is to address traffic oriented performance requirements, such as delay, delay variation, packet loss, and throughput, while ensuring that network resources are used economically and reliably. A major difficulty in achieving this goal lies in inconsistency of optimization objectives. Optimization objectives may change over time as new requirements appear, as new technologies emerge, and as new insights on the underlying problems are reached. Also, different networks may have different optimization objectives depending on their business models, capabilities, and operating constraints. So, the process of network performance improvement is not a one time goal. It is a continual and iterative process. To be prepared for possible changes of optimization parameters in time it is safest to have a flexible routing solution. That is, it is safest to look for a solution which imposes minimal limitations on traffic control in general.

The existing routing solutions have a limited set of options for traffic control purposes. Some of the imposed limitations have not yet been examined, and whether they can be avoided and the exact benefit of their removal is still unknown. This thesis examines traffic control options available in two routing solutions in the Internet. The first solution is the shortest path hop-by-hop routing deployed with the OSPF or IS-IS routing protocol and the IP forwarding protocol. This (IP) is the initially deployed and still the most popular routing solution in the Internet. The second solution is the explicit routing implemented with the RSVP-TE or CR-LDP signalling protocol and the MPLS forwarding protocol. This is the latest solution to have become widely deployed in the Internet. The objective of the research is to identify major limitations in the examined routing solutions so that techniques and tools which minimally limit traffic control in packet-switched networks can be developed. The focus is on techniques applicable in intra-area Internet routing. As there are advantages and disadvantages in both examined routing solutions, a potential third routing solution is to be considered which could draw from strengths of these two existing solutions.

The most popular routing approach in the Internet is shortest path routing. When networks implement this routing approach traffic is routed exclusively along the shortest

network paths. This is beneficial as it conserves network resources, but it does not necessarily ensure that network resources are maximally utilised. There are cases when the capacity of some shortest paths is insufficient for the traffic demands on the paths, leading thus to congestion on these paths. Alternative paths may then exist, which are not used, with sufficient capacity to take over some traffic on these critical shortest paths so that the congestion can be avoided. Solutions to enable use of such alternative paths so that network throughput is increased and congestion is avoided are considered in this thesis.

1.2 Contributions

The contributions of the research presented in this thesis are as follows.

- A common basic framework to describe existing routing solutions is identified and described in the thesis. When different routing solutions are compared using that framework weaknesses in these solutions are highlighted that are otherwise hard to spot.

- The major limitations which two popular routing solutions impose on network providers in controlling network traffic are identified in this thesis. The thesis provides a deeper understanding of the significance of these limitations and yields new insights that can guide the analysis and design of protocols involved in traffic control in packet-switched networks.

- Modifications of the existing protocols are recommended which would allow for a range of new traffic control approaches to be deployed in packet-switched networks. In particular, it is advocated that a number of routing and forwarding topologies of any type should be used when specifying traffic routes, with no constraints regarding the type of traffic in each topology.

- Two new routing topology algorithms based on these principles are proposed in this thesis. These algorithms define a set of routing topologies to be used in a single routing state and assign network traffic to routes available in these topologies so that network resources are efficiently and fairly used. Both algorithms base

their calculations on the physical network topology and traffic demands expected between network nodes. But the applicability of the two algorithms is different. The first algorithm, called the Link Mask Topology (LMT) algorithm, maintains traffic exchanged between any two network nodes on a single route exclusively. The second algorithm, called the Link Cost Topology (LCT) algorithm, distributes traffic exchanged between every two network nodes onto a number of routes, one in each routing topology it creates.

- A key performance objective of the LMT and LCT algorithms is to simultaneously achieve high network throughput and fairness in resource allocation. The performance of the two proposed algorithms is evaluated in the thesis in a number of test cases through calculations comparing them to the shortest path routing algorithm. The test results show potentials for the two proposed algorithms to considerably improve the performance of networks.

1.3 Thesis Outline

The remainder of this thesis is organized as follows.

Chapter 2 describes two existing routing solutions for packet-switched networks. The first solution is the shortest path hop-by-hop routing deployed with the OSPF or IS-IS routing protocol and the IP forwarding protocol, and the second solution is the explicit routing implemented with the RSVP-TE or CR-LDP signalling protocol and the MPLS forwarding protocol. For brevity, the two solutions are referred to as hop-by-hop IP routing and explicit MPLS routing, respectively, in the thesis. The limitations of these routing solutions are described in this chapter, and research directions for their solutions.

Chapter 3 presents a common framework to describe routing solutions for packet-switched networks. The presented framework is identified by the author of this thesis in an in-depth analysis of existing routing solutions. Understanding of the core operation of routing solutions in general is the crucial first step in the search for efficient traffic control techniques in packet-switched networks.

Chapter 4 describes the two existing routing solutions examined in the thesis, hop-by-hop IP routing and explicit MPLS routing, according to the identified common framework of routing solutions for packet-switched networks. Viewing of existing routing solutions from different perspectives helps in gaining better understanding of their operation and the limitations they have, which is the next major step in the search for efficient traffic control techniques in packet-switched networks.

Chapter 5 summarizes the recommendations of the author of this thesis in designing an improved routing solution for packet-switched networks. This chapter introduces two routing topology algorithms designed by the author of this thesis. They are called the Link Mask Topology (LMT) algorithm and the Link Cost Topology (LCT) algorithm. The motivation for the two algorithms was the potential inefficiency in network resource utilisation of using shortest path routing. The algorithms aim to maximize network throughput while maintaining fairness in the distribution of network resources.

Chapter 6 presents results of a set of performance tests carried out for the two routing topology algorithms introduced in the thesis by the author of the algorithms.

Chapter 7 gives a summary of the work presented in this thesis and concludes the thesis.

Chapter 2

Existing Routing Solutions

The largest existing packet-switched network, the Internet, is divided into Autonomous Systems. Each Autonomous System (AS) is a collection of networks under the control of a single authority. Larger Autonomous Systems are typically divided into multiple areas, as shown in Fig 2.1, which are relatively small in size. According to a survey the number of routers in one area ranges from 20 to 350 with 100 being the median and 160 being the mean [6]. Traffic in the Internet is in the form of IP packets. The network delivers them to their destinations. The responsibilities to carry out tasks required in achieving this are divided between human operators and network protocols, performed by routers. The most commonly deployed routing solution within an area, called *hop-by-hop IP routing*, and another which is widely deployed, called *explicit MPLS routing*, are described in this chapter.

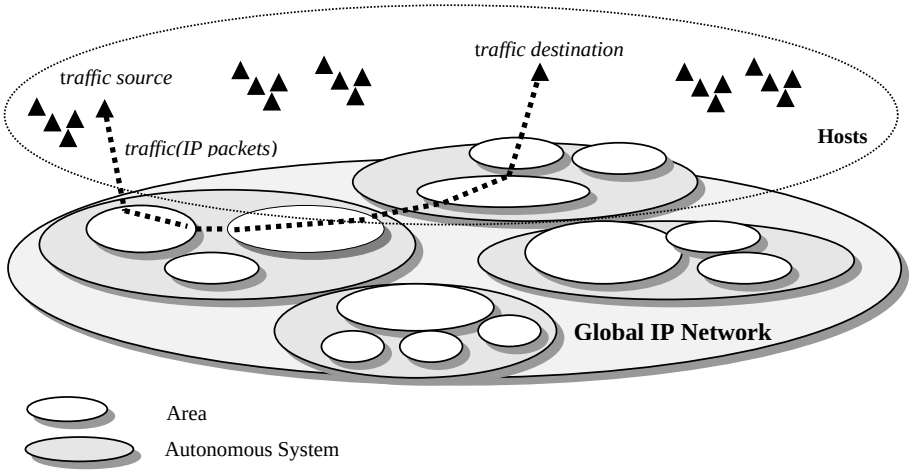


Figure 2.1: Structure of the Internet

2.1 Routing Information

A simplified area network example is shown in Fig. 2.2. To be able to route IP packets through the network, an IP router has to know the *base network topology* and which *addressing spaces* (*IP prefixes*) are reachable from which edge routers (edge routers in a network are routers which communicate with routers outside that network or with hosts). This information is referred to as *routing information*. It is provided to routers by human operators and by a link state routing protocol, either Open Shortest Path First (OSPF) [7] or Intermediate System-Intermediate System (IS-IS) [8].

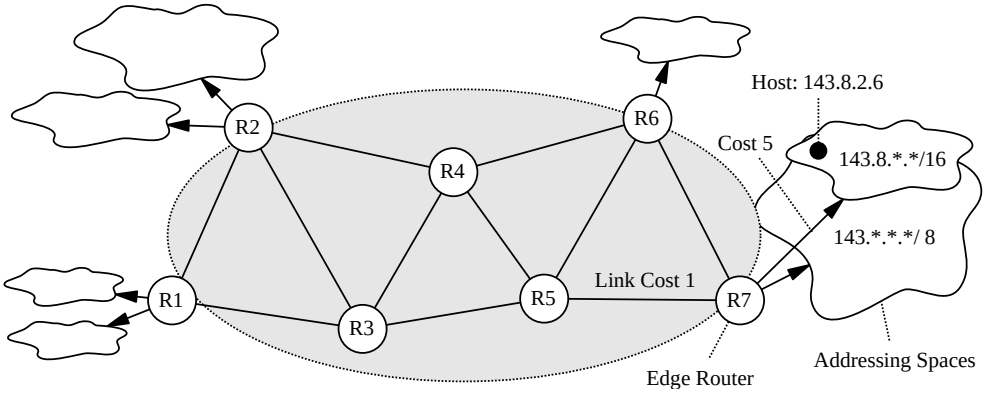


Figure 2.2: Area Network Example

2.1.1 Network Topology

The base network topology is defined by the set of network routers and links, and the link costs. The cost of a link is a metric which is assigned to the link, commonly by human operators but also possibly by automated tools. It can be regarded as the distance between routers connected by the link, whose value may be determined, for example, based on the link delay, capacity, or load on the link. Most commonly, the link cost is set to be inversely proportional to the link capacity, as suggested by Cisco [9], which ensures that higher capacity links which deliver packets with less delay have lower costs. In the OSPF and IS-IS standards, two byte protocol fields are reserved for the link costs. Routers and links have identifiers that are 4 bytes long. A router commonly has an identifier which is the same as the identifier of one of its links. The base topology information is provided to IP routers by human operators. The routers maintain it in the Topology Database (Fig. 2.4).

The network topology changes when a router or a link fails or when it is restored. To detect topology changes, neighbouring routers exchange *hello* messages at regular time intervals. A lack of a *hello* message from a neighbouring router is an indication that the neighbouring router and/or the link to that router has failed. The recovery is indicated by the next arrival of a *hello* message. Routers record such occurrences as different states of the corresponding links, i.e. as non-active and active link states, respectively. All routers advertise the states of their links in *link state* routing protocol messages periodically and when the link states change. The information provided in the link state messages received from other network routers is maintained at each router in the Routing Information Base (RIB) (Fig. 2.4). By examining current link states routers infer the *feasible* network topology, i.e. the network topology which contains currently active routers and links only.

2.1.2 Addressing Spaces

Addressing spaces which can be reached from an edge router are distinguished by their *IP prefix* addresses. The format of an IP prefix is $\{$*IP address/mask*$\}$. The mask specifies which bits in the given 4 byte long IP address define the addressing space [10]. These are consecutive bits starting from the first bit. An example is shown in Fig. 2.3. The standard method of writing an IP address is as the decimal value of each of the four bytes in the address, separated by full stops. The mask can be represented with the same notation, or by the number of bits which define the network (Fig. 2.3). Exceptions exist [10]. Hosts in an addressing space have 4 byte long IP addresses of the format $\{$*IP Prefix.Host*$\}$, e.g. in the addressing space whose IP prefix is 143.8.*.*/16, hosts have IP addresses 143.8.*.*. IP prefixes cover arbitrary territories which may overlap, as shown in Fig. 2.2. Hence a

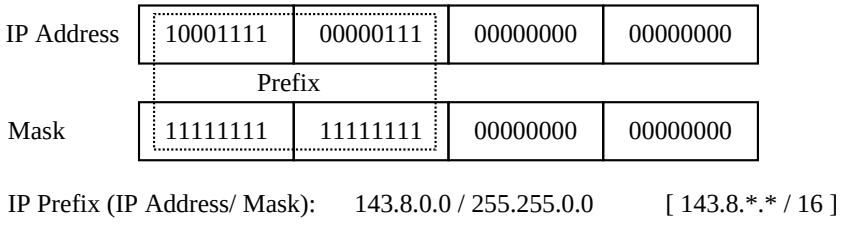


Figure 2.3: *IP Prefix* Address

host can belong to more than one addressing space. In such cases it is considered that

the host belongs to the addressing space whose IP prefix is the longest, as that prefix has the most precise address information. In the example given in Fig. 2.2 host 143.8.2.6. belongs to addressing space 143.8.*.*/16 because its IP prefix is longer (16 bits) than the IP prefix of addressing space 143.*.*.*/8 (8 bits).

Edge routers advertise reachable IP prefixes in *link state* routing protocol messages. There is a cost metric associated with each IP prefix which is reachable from an edge router. It can be viewed as the distance between the edge router and the addressing space identified by that IP prefix. This metric is set by human operators, or by edge routers based on the *external routing information*, i.e. based on the routing information provided by link state routing protocols in neighbouring area networks or by the path-vector routing protocol Border Gateway Protocol (BGP) [11]. BGP is an inter-area routing protocol, i.e. it distributes routing information between Autonomous Systems. The set of IP prefixes which an edge router can reach may change, for example, due to topology changes in other area networks. Each edge router detects this by examining the external routing information it has, and it advertises detected changes in link state messages network wide.

2.2 Hop-by-hop IP Routing

In hop-by-hop IP routing, IP routers make routing decisions independently by following the same strictly defined rules. An IP packet which traverses a network is routed independently at each *hop*, hence the name *hop-by-hop* routing. An alternative common name for this routing solution is *shortest-path* routing, because each IP router commonly routes packets along shortest paths to their destinations.

2.2.1 Protocols Used in Hop-by-hop IP Routing

The protocols required in hop-by-hop IP routing are a link state routing protocol, i.e. OSPF or IS-IS, and the forwarding protocol IP [12]. They are organized as shown in Fig 2.4. Their responsibilities in IP traffic control are as follows.

The routing protocol determines the current states of network links and advertises them periodically and when they change network wide. It also determines which edge

IP routers can reach which IP prefix and advertises that information network wide (section 2.1). Based on all routing information which is available at each IP router, this protocol establishes IP routing tables with forwarding instructions for IP.

IP forwards packets in compliance with the forwarding instructions which the routing protocol provides. It is also responsible for fragmenting packets whose size does not comply with the maximum packet size (Maximum Transmission Unit (MTU) [13]) which can be accepted on a link, and to subsequently reassemble those packets.

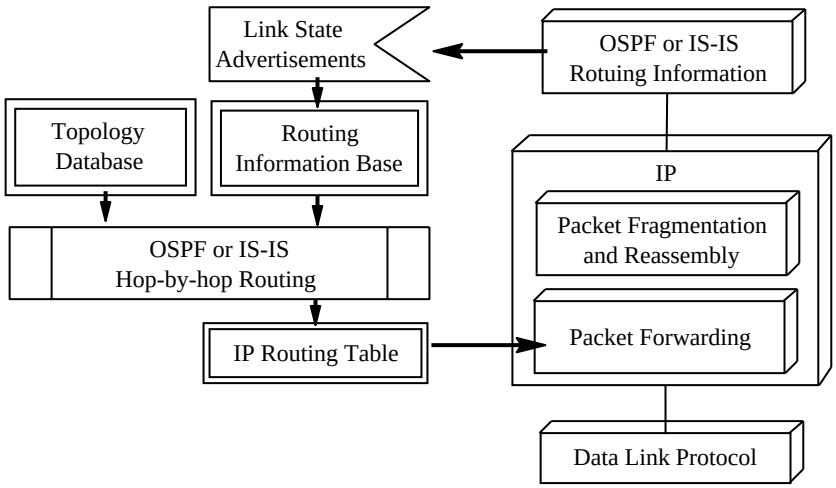


Figure 2.4: Protocol Stack in Hop-by-hop IP Routing

In its operation IP relies on feedback about the problems in the network provided by the Internet Control Message Protocol (ICMP) [14]. ICMP is considered to be an integral part of IP, though it operates *on top* of IP. Since its role in traffic control is minor when compared to the other two protocols, its operation will not be discussed here.

2.2.2 Shortest Path Algorithm

Forwarding paths in hop-by-hop IP routing are the shortest paths in the feasible network topology which are determined by routers using a shortest path algorithm. Most commonly that is Dijkstra's algorithm [15]. In general, Dijkstra's algorithm solves the single-source shortest path problem for a directed graph with nonnegative edge weights, i.e. costs. In an area network whose topology (the set of routers and links and the link costs) is known, Dijkstra's algorithm is used to determine the shortest paths, i.e. the minimum

cost paths, from a *source* router (s) to every other network router. The algorithm's operation can be described as follows.

The input of the algorithm is the network topology. It is denoted by $T(R, C_L)$, where R is the set of routers in the network, and C_L is the set of link costs. The cost of link $l(i, j)$ between routers R_i and R_j is a positive value. It is denoted by $c(i, j)$, where$(c(i, j) \in [0, \infty], \forall i, j)$.

The algorithm works by recording for each router v, the cost $c(v)$ of the shortest path found so far between the source s and v, and the previous hop or predecessor router on the path $p(v)$. This information defines the currently established *shortest path (SP) tree* from s through the network. The algorithm maintains two sets of routers, S and Q. Set S contains routers currently on the SP tree, while set Q contains routers not yet added to the tree. Initially, set S is empty, and $c(v)$ is set to 0 for s ($c(s) \leftarrow 0$), and to ∞ for every other router, since the costs of paths to these routers are unknown ($c(v) \leftarrow \infty, \forall v \in Q, v \neq s$). There is no previous hop information, hence $p(v) \leftarrow 0, \forall v \in Q$. The router with the minimal $c(v)$, that is s, is then added to the SP tree, i.e. it is shifted from set Q to set S, and new $c(v)$ values are calculated. If there is a link from router u on the SP tree (in set S) to router v which is not on the SP tree (in set Q), a *temporary* $c(v)$ is determined as $c(v) \leftarrow c(u) + c(u, v)$. If the temporary $c(v)$ is lower than the current value of $c(v)$, the current value of $c(v)$ is replaced with the temporary $c(v)$ and the previous hop for v is set to be u ($p(v) \leftarrow u$). The router with the minimal $c(v)$ in set Q is then shifted to set S, which results in router v and link $l(p(v), v)$ being added to the SP tree. This process is repeated until set Q is empty, i.e. until all network routers are on the SP tree.

A high level view of the algorithm operation is given in Desc.1. To illustrate its operation, the SP tree establishment in the topology shown in Fig. 2.5 when the source router is $R1$ is shown in Fig. 2.6.

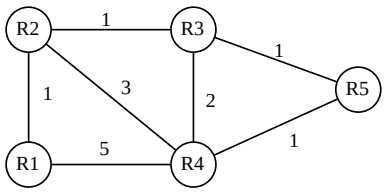


Figure 2.5: Network Topology Example

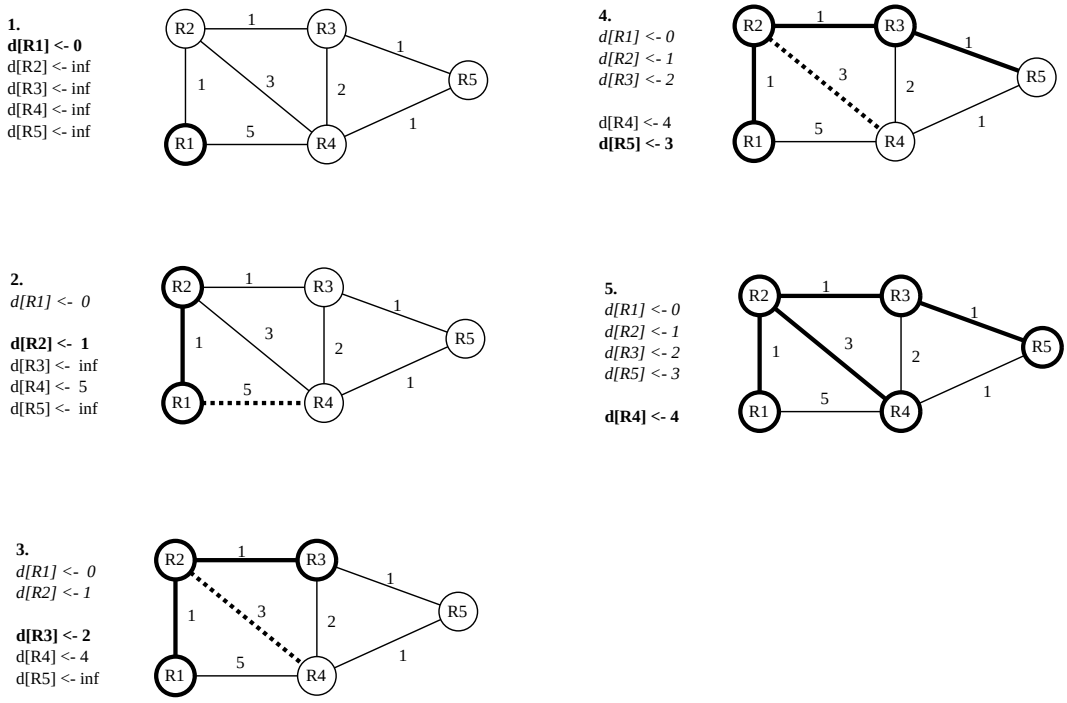


Figure 2.6: Shortest Path Tree Establishment for Router R1

The running time of the simplest implementation of Dijkstra's algorithm is $O(n^2)$, where n is the number of routers in the network. When the number of links per router is significantly lower that the number of routers in the network, i.e. in sparsely connected networks, this time can be improved to $O((m+n)\log n)$ by using binary heaps [16], and to $O(m+n\log n)$ by using Fibonacci heaps [17], where m is the number of links in the network. In recent years impressive speed-up techniques for Dijkstra's algorithm have been developed [18, 19, 20, 21, 22].

Desc.1: Dijkstra's Algorithm	
INPUT:	Network Topology $T(R, C_L)$ Source Router s
OUTPUT:	Shortest Path Tree for s $SPT(s)$

PROCEDURE:	$S \leftarrow 0; Q \leftarrow R; p(v) \leftarrow 0; c(v) \leftarrow \infty, \forall v \in Q;$
	$c(s) \leftarrow 0;$
	1. Determine minimal $c(u)$ and shift u from Q to S.
	2. Determine new $c(v), \forall v \in Q$:
	if $(c(v) > c(u) + c(u,v))$ then {
	$c(v) \leftarrow c(u) + c(u,v);$
	$p(v) \leftarrow u;$
	}
	3. Repeat steps 1 to 2 while $Q \neq 0$.

2.2.3 IP Routing Tables

Each IP router in an area independently determines paths through the network to every reachable IP prefix, i.e. to every reachable *external destination*, based on the routing information available to it, and stores the results in its IP routing table. In the process of building the IP routing table the following two steps can be identified.

In the first step an IP router determines the feasible network topology based on the information in its routing information base (RIB) (section 2.1), and the shortest paths in that topology from itself to all IP routers in the network, i.e. to all *internal destinations*. For each path the IP router records the identifier of the destination router and the identifier of the next link on the path, and the path cost. This information is shown in Table 2.1, which is referred to as the Network Paths table.

Table 2.1: Network Paths

Internal Destination	Shortest Path	Distance
Area Router ID	*Next Link ID*	*Cost*

In the second step, the router determines the shortest paths to all IP prefixes, i.e. to all external destinations. To determine these paths the IP router first examines its routing information base (RIB) to find out which edge network routers have advertised which IP prefixes as reachable and with what cost metrics (section 2.1). This information is shown in Table 2.2, which is referred to as the Reachable External Destinations table.

Table 2.2: Reachable External Destinations

External Destination	Advertised by Edge Router	Distance
IP Prefix	*Area Router ID*	*Cost*

Based on the information in Table 2.1 and Table 2.2, the IP router determines the costs of paths to IP prefixes, and the next links on these paths, as follows. The cost of a path to an IP prefix in Table 2.2, is the sum of the cost given in Table 2.2 for that IP prefix, and the cost of the network path in Table 2.1 to the edge router which has advertised that IP prefix in Table 2.2. The next link on the path is the next link on the network path to that edge router which is given in Table 2.1. The IP router records the IP prefix, the relevant next link and the total cost in Table 2.3, which is referred to as the External Paths table. When an IP prefix is advertised as reachable by more than one edge router, more than one

path exists to that IP prefix. In such cases only the path with the lowest total cost, i.e. the shortest path, and the next link on that path are recorded in Table 2.3.

Table 2.3: External Paths

External Destination	Shortest Path	Distance
IP Prefix	*Next Link ID*	*Cost*

Based on the information in Table 2.3, the IP router then creates an IP routing table of the format shown in Table 2.4. It contains all reachable IP prefixes. For each IP prefix, the identifier of the next link on the shortest path to that IP prefix is provided.

Table 2.4: IP Routing Table

External Destination	Next Link
IP Prefix	*Next Link ID*

The IP routing tables are updated whenever the content of Table 2.1 and Table 2.2 changes. When the routing protocol reports that the states of some network links have changed, new network paths are determined and Table 2.1 is modified. Table 2.2 changes when edge routers advertise that the connectivity to some IP prefixes has been lost or restored.

2.2.4 Packet Forwarding

After the IP routing tables are established at each router, as described in section 2.2.3, packets are forwarded through the network as follows. Each IP packet carries its IP destination address in its IP header (section 2.2.11) which the source of the packet creates and adds to the packet. Each router reads this address for every IP packet, and determines the longest IP prefix in its IP routing table which matches the address. It then forwards the packet to the next link associated with the determined IP prefix in its IP routing table.

The longest prefix IP routing table lookup which IP routers perform in the forwarding process is complex. The time required for such a table lookup used to be a major bottleneck of IP routing. However advanced lookup schemes have subsequently been developed which are sufficiently fast for link speeds of several Gigabit/sec [23, 24], and so sufficiently fast for current needs.

2.2.5 Congestion Control

Congestion in a packet-switched network occurs when a link or node is carrying so much data that its quality of service deteriorates. The packets may then be delayed or lost, and new connections may be blocked. In order for these negative effects to be avoided, congestion control techniques are applied.

The level of TCP traffic on forwarding paths in IP networks is controlled with congestion control techniques which inform sources of packets to reduce the transmission speed. Network routers request a lower transmission speed when they detect that too high a level of traffic is routed over their links given the links capacities. Traditionally, TCP/IP networks signal congestion by dropping packets. This concept has subsequently been replaced with ECN [25] congestion control, which allows end-to-end notification of network congestion without dropping packets. ECN is only used when both endpoints signal that they want to use it. When ECN is successfully negotiated, an ECN-aware router may set a bit in the IP header instead of dropping a packet in order to signal the beginning of congestion. The receiver of the packet echoes the congestion indication to the sender, which must react as though a packet drop were detected. ECN uses two bits of the TOS field in the IP header.

The need for congestion control in the Internet, what constitutes correct congestion control, and the dangers of neglecting to apply proper congestion control are explained in [26, 27].

2.2.6 Path Restoration

In hop-by-hop IP routing forwarding paths are modified when a network router or a link which is on a shortest path fails and when it is restored following a failure. As described in section 2.1, routers detect that a neighbouring router or a link to that router has failed or is restored, based on *hello* messages, which the neighbouring router sends periodically. Every router which detects that the state of one of its link has changed (from non-active to active and vise versa), advertises that in a *link state* message network wide. Upon receiving such a message, routers determine the feasible network topology, and calculate new shortest paths for the feasible topology, i.e. they update Table 2.1. Subsequently,

they update the IP routing table (Table 2.4) as well.

The feasible topology differs from the one for which the shortest paths had previously been determined in the number of routers and/or links, and thus it has a different set of shortest paths. The paths between some routers remain the same, but some paths are modified - it can be said that *a new set of forwarding paths is established*. This *path establishment process* ends after a link state message generated by the router which has detected a topology change reaches the most distant router, and after that router determines its shortest paths in the new feasible topology. It can be substantially prolonged when topology changes are frequent, because the path revisions triggered by the last change of link state will not have been completed before link state changes again. During this time routers operate with inconsistent topology information. The shortest paths they calculate are inconsistent as well, and packets may then be rerouted at each hop from one path to another. Since inconsistent paths may form a loop, some packets can thus end up looping for a while through the network. To prevent this, the number of hops a packet can take is limited. Each packet has a one byte long Time To Live (TTL) field in its IP header where the number of remaining hops it can take is maintained. This field is set by the source of the packet, and decremented in the forwarding process by IP at each hop. If zero is reached, the packet is discarded.

2.2.7 Path Protection

Recently a number of solutions has been proposed to improve the performance of hop-by-hop IP routing following topology changes by providing pre-established *backup* paths in additional *virtual* network topologies [28, 29, 30, 31, 32, 33]. These so-called *backup* topologies are defined centrally, based on knowledge of the base (*real*) network topology either by excluding links or by modifying link costs in the base topology. The shortest paths in these topologies are the backup paths which should be used for forwarding packets in case a *primary* path in the base topology fails. Studies have shown that from three to six topologies were necessary to achieve full fault tolerance (in the tested topologies) for link [31] and router [29] failures.

Means for distributing backup topology information exist in early versions of OSPF [7].

Recently extensions for the routing protocols OSPF and IS-IS have been proposed for so-called *multi-topology routing* in [34] and in [35], respectively. *OSPF* and *MT-OSPF* allow links to have multiple link costs, one for each topology. *M-IS-IS* reserves one bit *mask* for each link in each topology. The mask of a link in a topology indicates if the link exists or not in that topology. After the link costs or masks are defined for the backup topologies by human operators or by an integrated network management system, and set at each router, routers advertise the states of their links in all topologies, i.e. in a number of backup topologies and in the base topology. Subsequently all routers determine the shortest paths in all topologies and establish either a separate IP routing table for each topology, or an IP routing table with a number of *next link* entries, one for each topology, as is shown in Table 2.5. A set of backup paths is thus established network wide, in addition to the primary paths, to every external destination, i.e. to every reachable IP prefix. In this case traffic can simply be switched to a backup path, when a primary path fails, and thus the potentially lengthy path establishment process discussed in the previous section, is avoided. This is called path protection.

Table 2.5: *Multi-Topology* IP Routing Table

External Destination	Next Link, Topology 1	Next Link, Topology 2
IP Prefix	*Next Link ID*	*Next Link ID*

It is not specified in [34, 35] how packet forwarding should be performed in *multi-topology networks*, although it is claimed in [35] that this technique is used by many Internet Service Providers (ISPs) today. M-IS-IS suggests using different topologies for different destination addresses, or using the Type-of-Service (TOS) field in the IP header (section 2.2.11) as the topology number.

2.2.8 Equal-Cost Multi-Path Routing

There may be more than one shortest path to an external destination. In hop-by-hop IP routing there is an option for IP routers to split traffic among such paths by using equal-cost multi-path (ECMP) routing [36]. If packets belonging to a single session (i.e. packets sent by a single application from a source to a destination) are routed differently, the transport protocol [37] may have some difficulties in calculating round-trip delay and

the maximum packet size along the packets' path. The number of packets which are delivered out of order is increased as well. To prevent this IP routers take care to split only traffic from different sessions among different paths, i.e. they send packets of a particular session always onto the same path. Packets in a single session have the same *IP source and destination address*, and the same value in the *protocol* field of their IP headers. Hence, in ECMP routing, routers also maintain the information shown in Table 2.6 and Table 2.7. The square brackets in Table 2.7 indicate that the router can have some locally defined short identifiers which unambiguously define each session, and which can be determined by examining the fields in the brackets in the IP header of each IP packet.

Table 2.6: IP Routing Table with Multiple Equal-Cost Shortest Paths

External Destination	Next Link, Shortest Path 1	Next Link, Shortest Path 2
IP Prefix	*Next Link ID*	*Next Link ID*

Table 2.7: ECMP Routing Table

Session	Next Link
[*Source*, *Destination*, *Protocol*]	*Next Link ID*

The number of forwarding instructions in ECMP routing is clearly substantially higher than in *traditional* hop-by-hop IP routing. A justification for its deployment is that by splitting traffic along equal-cost shortest network throughput can be increased.

2.2.9 Optimizing Link Costs

Using only the shortest paths as the forwarding paths in a network may mean that the network handles less traffic than it could, given its resources and the current traffic demands. Congestion may happen despite the presence of under-utilised non-shortest paths in the network which are not used, but which could take over some traffic on the shortest paths so that the congestion is avoided. To increase network throughput and lower the probability of congestion it has been proposed to optimize link costs based on a network wide view of traffic and network topology so as to arrive at the best set of shortest paths to carry a particular pattern of demands, i.e. the one that would enforce an approximately even traffic distribution on network links. A number of schemes for determining link costs based on fixed known demands have been experimentally evaluated on real networks, showing that

it is possible to find a set of link costs which allows significantly more demands (50%-110%) to be supported in the network when compared to the Cisco's inverse-capacity costs [38, 39, 40, 41, 42, 43, 44]. Some schemes aim for a link setting that performs well also in the presence of link failures [45, 46, 47]. However, it is relatively simple to find demands and a topology where the performance of this scheme is not satisfactory [39].

The *control* of link costs is intended to be performed rarely, for example, when the network topology is extended or when the demands change significantly [48]. In general, an integrated network management system could automate the entire process of detecting congestion, selecting suitable costs and effecting the configuration changes. There are however some major issues with modifying link costs, since such changes will lead to a period of routing instability as the routing protocol converges on the new topology [49]. Every single link cost change has to be reported to all routers. After being informed about the new costs, routers have to calculate shortest paths and to update their IP routing tables. So it may take seconds before the shortest paths defined by a new set of costs are established. In the meantime packets are routed in an unpredictable manner and may consequently be delivered out of order or be lost. This is particularly critical when a number of new link costs needs to be advertised network wide [48].

A link cost can also be a quantity that varies with the link utilisation. While a fixed cost needs to be distributed only when a link goes down or when it recovers, variable costs need to be distributed more frequently, periodically, or by using triggered messages. The routing protocol overhead is thus increased. In contrast to periodic updates, triggered messages complicate the provisioning of network resources since rapid fluctuations in available capacity can generate a large number of link-state updates, unless a reasonable hold-down timer is used [50]. Frequent link-state updates may lead to undesired traffic fluctuations in the network. If all network routers are informed that a link is overly utilised, they will route traffic away from the link. This will however drastically reduce the utilisation of that link and it will then be advertised as a low-utilised link. All the rerouted traffic will then be shifted back to the link. The link may then again become overly utilised and the cycle starts again [51].

2.2.10 Providing More Forwarding Paths

When network links on shortest paths have insufficient capacity for all traffic which is routed along these paths, it may happen that longer paths exist with sufficient capacity to *take over* some traffic on the shortest paths. However, IP routers record routing instructions for shortest paths only. This implies that inefficient resource utilisation in IP networks primarily occurs because there is a low number of available forwarding paths.

The number of forwarding paths in IP networks can be increased by defining a number of virtual network topologies, which have different sets of routers, links and/or link costs, and thus different sets of shortest paths as well, and by enabling routers to determine shortest paths in each virtual network topology, i.e. by enabling *multi-topology* routing. When a number of paths is provided between network routers, the problem arises of how network routers assign traffic to one of the available paths in response to traffic dynamics, so that network resources are efficiently used. This traffic control solution has been neglected in research circles. Early works on this topic can be found in [52, 53, 54].

2.2.11 IP Header

The format of the IP header is shown in Fig. 2.7. It is at least 20 bytes long. The numbers at the top represent byte positions. The field definitions are given below. In general, only the fields marked in the figure are necessary in the packet forwarding process.

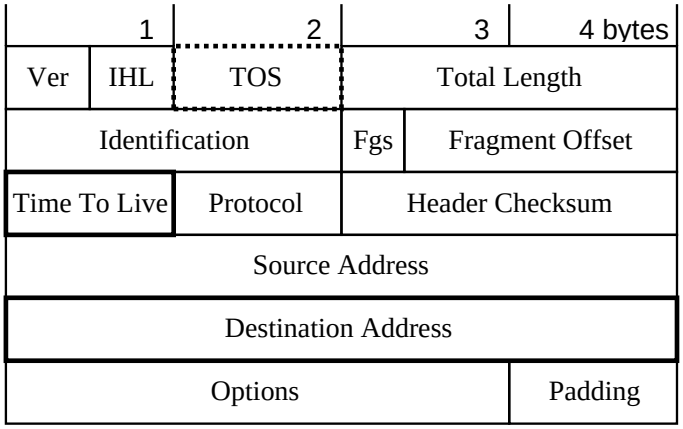


Figure 2.7: IP Header Format

- *Version* is the Internet Protocol version number. In IP which is considered here it is set to 4.
- *IHL*, Internet Header Length, is the length of the IP header in 32 bit units.
- *Type of Service* was originally intended to define parameters of the type of service desired. Currently 2 of the bits of this field are used in ECN [25] congestion control. The remaining bits are used by DiffServ [55].
- *Total Length* is the length in octets of the packet (fragment), including the IP header.
- *Identification* is a value assigned by the sender to be used in packet reassembling.
- *Flags* are control flags used in packet fragmentation and reassembly. One bit is reserved and must be set to 0. The other two bits are:

  DF - don't fragment, and

  MF - more fragments.
- *Fragment Offset* is the offset of this fragment in the original packet, in 64-bit units.
- *Time to Live* is the number of hops a packet may take on its way to a destination.
- *Protocol* determines the next level protocol used in the data portion of the IP packet.
- *Checksum* is a checksum performed on the header only.
- *Source Address* is the IP address of the source of the packet.
- *Destination Address* is the IP address of the destination of the packet.
- *Options* are optionally present, used for security, source routing and other functions.
- *Padding* is a zero padding to ensure that the IP header ends on a 32 bit boundary.

2.3 Type of Service

Traditionally in hop-by-hop IP routing all IP packets are regarded to be of equal importance. Network resources are fairly shared by packets from different sessions and when the network is overused, the service degrades equally for all customers. No guarantees regarding packet loss, delay and order of packet delivery are provided, i.e. hop-by-hop IP routing provides *best-effort* packet delivery only. In recent years a wide range of communication-intensive, real-time multimedia applications have appeared. Unpredictable delays and possible out-of-order packet delivery are undesirable for real-time traffic, so new requirements have been placed onto the network to meet various service requirements traffic can have. *The set of service requirements to be met by the network while transporting a flow* is referred to as *Quality of Service (QoS)* [2]. Routing of traffic flows while insuring that their QoS requirements are met is called *Quality-of-Service (QoS)* routing. A possible need for QoS routing was considered from the very beginning of IP network development. The *Type of Service* (TOS) field was reserved in the IP header (section 2.2.11) for this purpose. However techniques for using this field are still under development. In past two decades two models for QoS support in IP networks have been developed: Integrated Services (IntServ) [56] and Differentiated Services (DiffServ) [55]. There were also proposals for inter-connecting these two service models in [57, 58, 59]. It was expected that the resulting model would draw from the strengths of both models. Further research on this topic has led to the development of MPLS networks [60, 61, 62, 63] which are described in section 2.4.

2.3.1 Integrated Services

In the *Integrated Services* (IntServ) model in addition to the best-effort service two more service classes are defined. They are:

guaranteed service [64] - for applications requiring bounded packet delivery time, and

controlled load service [65] - for applications that can tolerate some delay and are sensitive to traffic overload condition.

The IntServ model is based on the following assumption: routers have to be able to reserve resources a priori for a traffic flow in order for a required QoS for that flow to be provided, which then further implies that routers have to maintain flow-specific states [56]. The additional components required for enabling IntServ in IP networks are: a signaling protocol (e.g. RSVP [66]), an admission control routine, a packet classifier and a packet scheduler. The role of the signalling protocol is to setup a path for a given traffic flow and to reserve resources along that path in accordance with the given QoS requirements before data transmission starts. The admission control routine decides whether the requested service can be granted. The classifier places packets into different service queues and they are then scheduled appropriately by the scheduler.

The IntServ architecture is computationally very demanding and it does not scale well. The amount of state information increases proportionally with the number of flows and this number can be tremendous in the IP network core.

2.3.2 Differentiated Services

The Differentiated Services (DiffServ) model aims to avoid scalability issues which exist in the IntServ model by aggregating traffic flows which require the same treatment, and thus reducing the flow-specific state information which routers have to maintain. These *flow aggregates* are then to be treated differently, according to the given requirements. A customer might have to register for DiffServ with its Internet provider network. A *Service Level Agreement (SLA)* [67, 68] is then reached between the customer and the network which defines classifier rules as well as metering, marking, discarding and shaping rules.

Examples of additional service classes beside the best-effort service in the differentiated services model are:

assured service [69] - for applications which require reliable transport, and

premium service [70] - for applications which require low delay and low jitter.

IP packets are classified, and possibly policed, at the ingress of the *DiffServ* network. Packets which require different types of service are distinguished by the value of the *Differentiated Services* (DS) field in the IP header [55]. The DS field is a 6-bit part of the TOS field. The ongoing research on DiffServ is extensive [71, 72, 73, 74, 75, 76].

2.4 Explicit MPLS Routing

When the lookup speed of IP routing tables was identified to be the major bottleneck in IP routing, proposals appeared to trade packet header for packet processing, i.e. to enable a new faster forwarding solution within independent IP network areas by deploying a new forwarding protocol which operates with a short packet header [77, 78]. Newly developed fast longest prefix lookup schemes have made the deployment of a new forwarding protocol less necessary. However, requirements to support QoS routing in IP networks have appeared. To fulfill these requirements it has been necessary to modify the current IP forwarding process. The proposal to deploy a new forwarding protocol has then been merged with the proposal to deploy a new signalling protocol for establishing paths and reserving resources along the paths to support QoS routing which was previously proposed in the IntServ model. A new forwarding protocol, called Multiprotocol Label Switching (MPLS) [60] and new signalling protocols, CR-LDP [61] and RSVP-TE [62], were developed and subsequently widely deployed in the Internet. The forwarding concept in MPLS networks to a great extent resembles cell switching across virtual paths which is deployed in ATM networks [79]. It is described in this section.

2.4.1 Terminology of MPLS

In the MPLS specification [60], a set of routers implementing MPLS that are in the same routing or administrative domain is called an MPLS domain. Routers which support MPLS are called *Label Switching Routers* (LSRs). LSRs that communicate with routers outside the MPLS domain or hosts are referred to as *Label Edge Routers* (LERs).

Traffic is routed through an MPLS domain along so-called *Label Switched Paths* (LSPs). Depending on the position of LSRs on LSPs in the domain, the following types of LSRs are defined:

- An *Ingress LSR* is the first LSR on an LSP in the MPLS domain and it handles IP packets as they enter the domain.
- An *Egress LSR* is the last LSR on an LSP in the MPLS domain and it handles IP packets as they leave the domain.

- A *Transit LSR* is an LSR between the first and the last LSR on an LSP in the MPLS domain and handles IP packets as they travel through the domain.

Hence any LSP starts with an ingress LSR and ends on an egress LSR. Ingress and egress LSRs are always edge routers, i.e. LERs. A transit LSR may or may not be an edge router.

MPLS is enabled in IP networks by enabling IP routers to support MPLS, in addition to IP which they already support. Hence each LSR is also an IP router. Provided that all IP routers in an IP network area support MPLS, that area network is an MPLS domain.

2.4.2 Protocols

In an MPLS domain it is the responsibility of a signalling protocol, either CR-LDP or RSVP-TE, to provide the forwarding protocol MPLS with the forwarding instructions by building the MPLS forwarding table at each router. When performing these tasks the protocol relies on the routing information provided by whichever routing protocol is deployed in the network (section 2.1). The signalling protocol is also responsible for monitoring the states of the established paths and for reporting detected failures to routers along the paths (e.g. by sending refreshing messages, section 2.4.6). Packets are forwarded through the network by MPLS in compliance with the decisions of the signalling protocol.

The interacting protocol stack in MPLS networks which operate with the signalling protocol RSVP-TE is shown in Fig. 2.8. CR-LDP operates on top of TCP [37], which operates on top of IP.

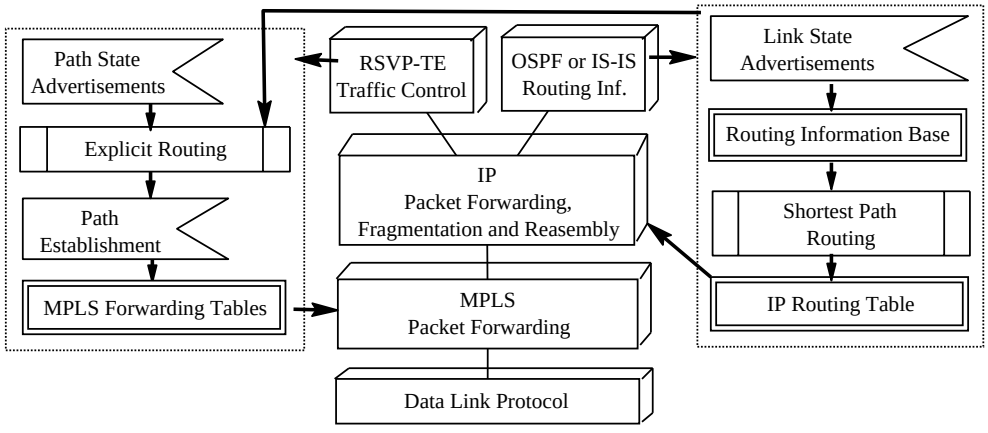


Figure 2.8: Protocol Stack in Explicit MPLS Routing

2.4.3 MPLS Forwarding Tables

Forwarding tables for LSRs in an MPLS domain are established as follows.

Incoming IP packets which enter the MPLS domain are classified by ingress LSRs into *Forwarding Equivalence Classes* (FECs). A FEC represents a group of packets that should be forwarded in the same way through the MPLS domain. The classification into FECs is done using packet filters that examine IP header fields such as source and destination IP address, Type-of-Service, etc.

Using a signalling protocol, each ingress LSR then initiates label assignment to each FEC in the domain down an explicit path determined by the ingress LSR. In this process each LSR along the path assigns a label, i.e. a short fixed value identifier, to the FEC and informs the predecessor router which label it has chosen. Labels have local significance. A label identifying one FEC in one LSR, may identify another FEC in another LSR.

Each LSR along the path stores the label it has assigned to the FEC, i.e the *incoming* label, the label assigned by the next LSR on the path, i.e. the *outgoing* label, and the identifiers of the previous and the next interface on the path in a *Next Hop Label Forwarding Entry* (NHLFE) table (Table 2.8). The ingress LSR stores the label assigned by the next LSR on the path and the identifier of the next interface on that path in an *Incoming Label Map* (ILM) table (Table 2.9). It also stores the FEC identifier and the label assigned to the FEC in a *FEC-to-NHLFE* (FTN) table (Table 2.10). The egress LSR records in its NHLFE table that it is the last router on the defined path.

The ingress LSRs can also initiate label assignment down the paths which are used in hop-by-hop IP routing [60]. This is called *hop-by-hop MPLS routing*.

Table 2.8: NHLFE Table

Incoming Interface	Incoming Label	Outgoing Label	Outgoing Interface
Previous Link ID	*Local FEC ID*	*FEC ID at Next LSR*	*Next Link ID*

Table 2.9: ILM Table

Label	Next Interface
FEC ID at Next LSR	*Next Link ID*

Table 2.10: FTN Table

FEC	Label
Local FEC ID	*FEC ID at Next LSR*

2.4.4 Packet Forwarding

Once the MPLS forwarding tables are created as described in section 2.4.3 (Tables 2.8, 2.9 and 2.10), IP packets are forwarded through the MPLS domain as follows.

Each ingress LER determines the FEC for each incoming IP packet and the label associated with it in its FTN forwarding table by examining the packet's IP header. It creates a short MPLS header, stores the label in this header and adds the header to the packet, *in front of* the IP header. The packet is then forwarded to the next interface associated with this label in the ingress LSR's FTN forwarding table.

Each LSR on the packet's path examines the label in the packet's MPLS header. The label determines the NHLFE forwarding table entry containing the record of where to forward the packet, and with which outgoing label. The LSR replaces the label in the MPLS header with the outgoing label recorded in the table and it forwards the packet to the next LSR on the path.

The egress LSR removes the MPLS header from the packet and then forwards the packet to a router in the next domain or to a host as an IP packet, i.e. it examines the packet's IP destination address and it follows the forwarding instructions provided in its IP routing table.

2.4.5 Label Switched Paths

A *Label Switched Path* (LSP), the path through one or more LSRs that packets in a particular FEC follow in an MPLS domain, is determined and established by the ingress LSR. The signalling mechanisms for LSP setup permit specification of QoS attributes for the LSP. The ingress LSR can set these attributes according to the demands regarding bandwidth, delay and/or packet loss which are specified, for example, for a particular type of traffic between a source and a destination in a customer Service Level Agreement (SLA) (e.g. streaming multimedia may require guaranteed throughput to ensure that a minimum level of quality is maintained, Voice over IP (VOIP) may require strict limits on jitter and delay, Video Teleconferencing (VTC) requires low jitter and latency). Packets of the specified type are then assigned to a single FEC, and they are forwarded along a *constraint-based* LSP for that FEC. This is commonly referred to as QoS routing.

An example of how an ingress LSR can determine a path through the MPLS domain which satisfies a bandwidth demand between a source, s, and a destination, d, follows. This demand is denoted by $b(s,d)$. To be able to determine a path which can support the demand $b(s,d)$, the ingress LSR has to be informed of the current residual bandwidth on each network link. The residual bandwidth on a link is the difference between the bandwidth of the link and the LSP demands that are routed on that link. It can be obtained from routing protocol extensions, such as in [80, 81, 82]. By pruning the links with insufficient residual bandwidth to support the demand from the base topology, the ingress LSR determines the *feasible topology for the demand*. This topology is called here the *constraint-based topology*. The shortest path to the egress LSR for the demand in the constraint-based topology, which is determined using Dijkstra's algorithm (section 2.2.2), is the path through the MPLS domain for that demand. The ingress LSR establishes it as an LSP using the signalling protocol and it reserves bandwidth $b(s,d)$ along that LSP.

This is the simplest *constraint-based path algorithm* [62] used in explicit MPLS routing. An outline of the algorithm operation is given in Desc.2. Aiming to improve the efficiency of explicit routing, other constraint-based path algorithms have been proposed [83, 84, 85, 86]. Some take more information into account when determining paths, such as the network ingress-egress points [87, 88, 89], and the estimated future demands for bandwidth on paths between the network ingress-egress points [90]. These algorithms differ from that described above only in how they determine the constraint-based topology, either by excluding links in the base topology or by modifying the link costs, prior to the calculation of shortest paths using Dijkstra's algorithm.

Desc.2: Constraint-Based Path Algorithm	
INPUT:	Base Network Topology $T(R,C_L)$ Residual Link Bandwidth B_L Bandwidth Demand $b(s,d)$
OUTPUT:	Constraint-Based Topology $CT(R',C_{L'})$
PROCEDURE:	Exclude all links l if $b_l < b(s,d)$ from $T(R,C_L)$ to generate $CT(R',C_{L'})$
INPUT:	Constraint-Based Topology $CT(R',C_{L'})$ Ingress LSR i
OUTPUT:	Shortest Path Tree for i $SPT(i)$ Path (i,e) between ingress and egress LSR for $b(s,d)$
PROCEDURE:	Dijkstra's Algorithm

2.4.6 Path Protection

An LSP has to be modified when a router or a link which is on the LSP fails. In such cases the signalling protocol informs the ingress LSR on the LSP which LSP has failed, and the ingress LSR then determines and establishes a new LSP to replace the failed one. In the MPLS domain which operates with RSVP-TE [62] the egress LSR on an LSP sends periodic *refreshing* messages to the ingress LSR on that LSP. The lack of these messages indicates to the ingress LSR that the LSP has failed. When the signalling protocol is CR-LDP [61] the transit LSR which has detected a failure of an LSP informs the ingress LSR on that LSP about the LSP failure.

After the ingress LSR has determined the new LSP, it establishes the LSP by sending a signalling protocol message to the LSRs on the new LSP. This LSP is established after the signalling message generated by the ingress LSR has reached the egress LSR on the LSP. If a link or a router on the new LSP fails during this process, all the above has to be repeated. Hence providing a new LSP might take a long time, particularly when topology changes are frequent. To avoid this a set of *backup* LSPs can be pre-established in the MPLS domain for the *primary* LSPs. In the 1+1 protection approach, traffic is sent simultaneously over the primary and the backup LSP. In the 1:1 protection approach the backup path is used only if the primary path fails, so that backup path resources may be shared by different backup paths that are activated in different failure scenarios [33].

When the backup paths already exist, traffic on primary paths can be rerouted faster following topology changes. However, when a primary path fails due to the failure of a link or a router somewhere along the path, the primary path failure has to be signalled first to the ingress LSR on the path, which should then reroute the traffic on this path to a backup path. This procedure takes some time. Thus fast rerouting requires a backup path starting at each router along the primary path [91], but such a solution entails tremendously many paths in the network. This can be reduced by a so-called facility backup option where each link or router failure is protected by separate backup paths. The facility backup deviates many primary paths at once around the failure locations. These paths are then reconnected after the failure locations.

2.4.7 Label Space

With the 20 bit long label which is used in explicit MPLS routing it is possible to distinguish up to 2^{20} label switched paths (LSPs) *at any single* label switched router (LSR) in an MPLS domain. Substantially more LSPs could be distinguished within the entire domain depending on the network topology. However, the number of LSPs which are established over an LSR equals the number of entries in its forwarding table. The more paths are established, the larger the forwarding table is. This has raised concerns about the table size requirements in explicit MPLS routing and has motivated proposals to minimize these requirements. Given that LSRs assign a label to each LSP that they are on, the number of labels an LSR uses is also equal to the number of entries in its forwarding table. The problem of minimizing forwarding table size is thus discussed as the problem of reducing the label space usage by label switched paths in [92]. Larger label space is also related to a longer lookup delay in [93] and to a lower network utilisation in [94].

The label space usage in explicit MPLS routing can be reduced by using multipoint-to-point egress routed *label switched trees* (LSTs) instead of LSPs. These trees can be defined by merging LSPs previously defined by some existing constrained-based path algorithm or independently based on the network topology and estimated traffic demands. Examples can be found in [95, 93, 96, 97].

2.4.8 Label Stack

When there is a number of MPLS domains in an MPLS network which are on different hierarchy levels, a packet may carry more than one MPLS header while traversing the network. The MPLS headers are then organized as last-in first-out stack called the *label stack* [98]. The number of label stack entries determines the stack depth. In an MPLS domain LSRs make forwarding decisions based on the labels of the same depth in the label stack. Processing of the labeled packet is always based on the label in the top MPLS header, and thus in such cases LSRs in an MPLS domain might have to perform additional stack operation in the forwarding process, e.g. they have to add an MPLS header to the stack or to remove it. The instructions for performing such operations are recorded in their forwarding tables when the tables are established. This mechanism can be used to

support tunnelling.

The label stack feature can also be used in connection with link failures. An LSR may push a label representing a detour around the failed link on the stack of packets which have previously been routed to that link [99]. LSRs in an MPLS domain then do not necessarily make forwarding decisions based on the labels of the same depth in the stack.

2.4.9 MPLS Header

The format of the MPLS header is given in Fig. 2.9. It is 4 bytes long. The numbers at the top represent bit positions. The field definitions are given below.

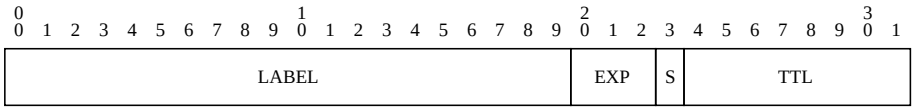


Figure 2.9: MPLS Header

- *Label*, a 20-bit field with the actual value of the label,
- *EXP*, a 3-bit field reserved for experimental use (used by DiffServ [55]),
- *S*, which indicates the bottom of the label stack (the packet payload immediately follows the label stack entry which has the S bit set),
- *TTL*, Time to Live.

The MPLS header is either encapsulated between the data link header and the network header (IP header) or uses an existing field in the data link header or the network header.

2.4.10 Time To Live

In the IP header the TTL field contains the number of hops a packet is allowed to take in the network. It is decremented at each hop and the packet is discarded if zero is reached. The field is a way of protecting against forwarding loops that may happen, for example, due to inconsistency of the routing tables following topology changes. Since within an MPLS domain no IP header examination is done, MPLS provides a way to keep the value of the field as it would be if the packet were IP routed. This is done by copying the TTL

field from the IP header of the packet into the TTL field of the MPLS header, as the packet enters the MPLS domain, and by decrementing it at each hop along an LSP. As the packet emerges from its LSP, the field is copied back to the TTL field of the packet's IP header. If the TTL value reaches zero somewhere along the path, the packet is discarded.

2.5 Tunnelling

Tunnelling or *encapsulation* is *handling of protocol A's packets, complete with A's header information, as data carried by protocol B* [100]. An encapsulated protocol A packet has a protocol B header, which is followed by a protocol A header, and then followed by the information that protocol A carries as its own data. Protocols A and B may be the same protocol. Two examples are shown in Fig 2.10. An IP packet is *tunnelled* through an MPLS domain as an MPLS packet. It can also be tunneled through an IP network area as another IP packet.

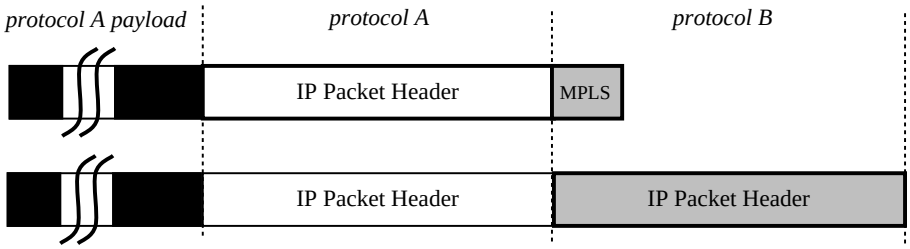


Figure 2.10: Tunneling or Encapsulation

2.6 Recommended Reading

More information on operations of packet-switched networks in general, and in particular on hop-by-hop IP routing, can be found in the excellent book by Radia Perlman called *Interconnections: Bridges, Routers, Switches and Internetworking Protocols* [100]. Grenville Armitage has summarized the key differences between hop-by-hop IP routing and explicit MPLS routing in the article *MPLS: The Magic Behind the Myths* [101]. More detailed descriptions of protocols mentioned in this chapter can be found at *http://www.ietf.org*.

Chapter 3

A Generalised Model of IP Routing

Comparing the two approaches to IP routing discussed in chapter 2 is complicated by the inconsistent terminology used in the literature to describe them. A common framework and terminology to describe IP routing is developed below.

3.1 An Overview of Traffic and Networks

A routing solution is the set of tasks which have to be performed so that traffic in a network can reach its destination. The network consists of nodes and edges. Traffic consists of traffic units. Traffic units transport their payload between specified destinations. They may have different priorities, in which case low(er) priority traffic units can be delayed or stopped in order to reduce delays of high(er) priority traffic units.

The traffic units in packet-switched networks are packets. Their payload is information. A large amount of correlated information can be sent to a destination in a continuous stream of packets. The network nodes are routers, the network edges are links, and destinations are hosts. Any type of connectivity between a host and a router is an access edge. A router is a container for information and protocols at each node. Protocols are software and hardware modules which perform operations required in the network. They are deployed in a layered architecture. The protocol(s) on layer k provides services for the protocol(s) on layer $k+1$. A packet header of level k is reserved for recording information that the protocol(s) on level k require. It encapsulates the packet and all of the packet headers of level $k+i, i>0$. In the layered protocol architecture, protocols on the network

protocol layer are responsible for routing and packet forwarding.

3.2 A High Level Description of Packet-Switched Networks

In packet-switched networks the set of destinations and the network characterise two functionally different strata. The destination stratum sets above the network stratum, as shown in Fig. 3.1. The requests for transport of some payload between specified destinations are generated in the destination stratum. The transport is performed in the network stratum. Connections between destinations and network nodes are called access edges, either ingress or egress. They form the interface between the destination and the network strata. The concept of a destination stratum allows addressing issues to be incorporated in the routing model.

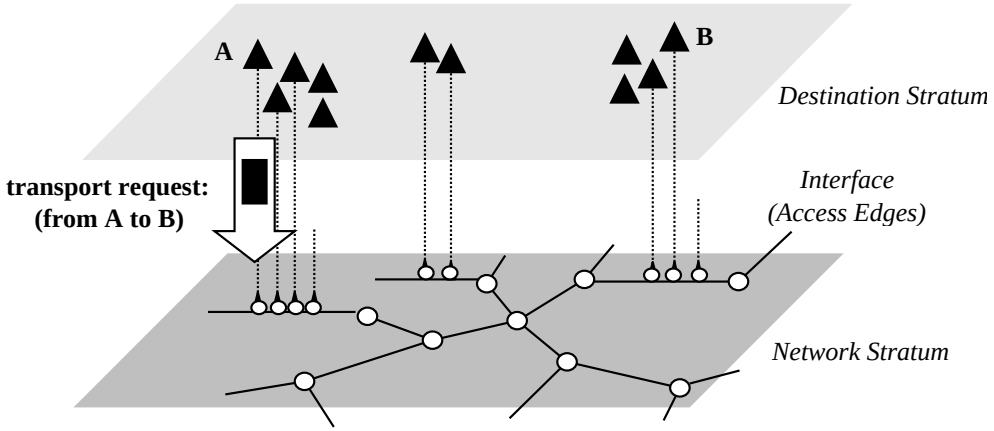


Figure 3.1: Destination Stratum and Network Stratum

3.2.1 The Destination Stratum

Destinations have hierarchical addresses. An example is shown in Fig. 3.2. There are hierarchical territorial divisions on the destination stratum into destination zones. A destination zone of level k has an identifier, e.g. *France*. There are smaller destination zones on a zone of level k which belong to level $(k-1)$. Their identifiers start with the identifier of the zone of level k that they are on, e.g. *Paris* in *France* has address *France.Paris*. Individual destinations are on level zero. The address of a destination starts with the identifiers

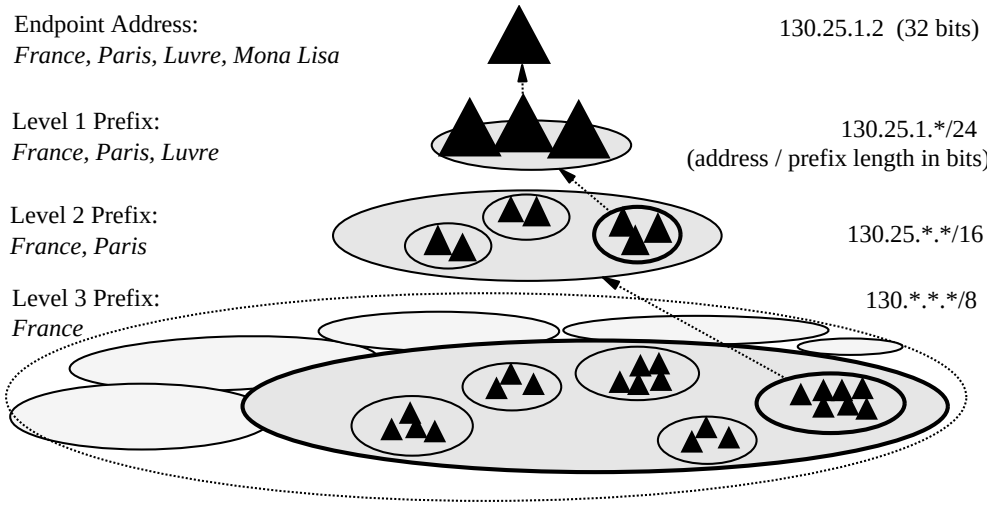


Figure 3.2: Hierarchical Destination Addresses

of the higher level destination zones that the destination is on, e.g. Mona Lisa's address is *France.Paris.Louvre.MonaLisa*. An address prefix is a notation used for identifying all destination addresses in a destination zone whose identifier appears in the address prefix, e.g. *France.** covers all destinations in *France*. The representation of destination addresses in packet-switched networks is numerical. An example is shown on the righthand side in Fig. 3.2, where sets of bits of different sizes (with 32 bits being the maximum in the example) are reserved for addressing destination zones on different levels. The addresses are binary numbers, but their decimal notation is given in the figure. Each four bit chunk of a binary number is represented with a decimal number.

3.2.2 The Network Stratum

The network is divided into areas which are controlled and maintained by different parties. There are hierarchical territorial divisions within the network stratum into network zones. They define polices on borders between network areas. On the border between two areas in a single network zone the policies of that zone apply. The border between two areas in two different network zones of level k is controlled by the policies of the network zone of level $k+1$ to which both belong. For example, each city and the intercity area in France are individual network areas in network zone *France*. *France* and *Germany* are two network zones in a higher level network zone *Europe*. *French policies* apply between the

network areas within *France*. *European policies* apply on the border between a network area in *France* and a network area in *Germany*.

In the largest existing packet-switched network, the Internet, the hierarchical territorial divisions on the destination level and the network level do not correspond to those presented on world maps, and used in the examples above. Instead, to use the same example, prefix *France* is missing in Mona Lisa's address.

The networks within single areas are small in size, in the approximate range of a hundred to two hundred nodes. They can be presented as shown in Fig. 3.3. The example in the figure is taken from [102]. In the example, a set of destinations, whose address prefix is *City**, is reachable over access edges from each node *City*, e.g. the addresses of destinations which are reachable from node *Vienna* start with prefix *Vienna**. The access edges are not shown in the figure. Each shown edge represents two unidirectional network edges. There are in total n nodes, m edges and d destinations. Each edge has a capacity of c.

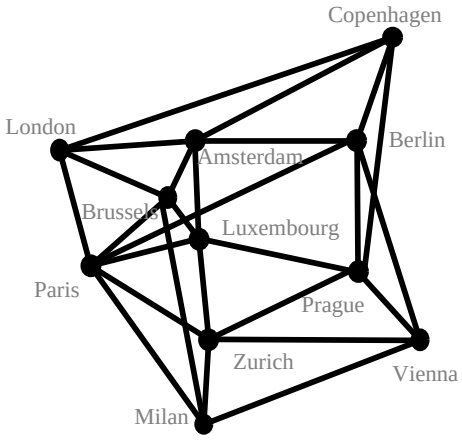


Figure 3.3: Network Stratum

3.3 Forwarding Topologies

Packets are forwarded through the network along forwarding topologies. A forwarding topology contains a set of unidirectional network edges, and a set of network nodes. Each node has at most one outgoing edge. That is the edge by which a packet which follows

the forwarding topology leaves that node. Three examples of forwarding topologies are shown in Fig. 3.4. Nodes with no incoming edges are called leaf nodes. Nodes with no outgoing edges are called root nodes. Topologies with a single leaf node and a single root node are called directed lines. Topologies with multiple leaf nodes and a single root node are called directed trees. Topologies with neither a root node nor leaf nodes are called directed rings.

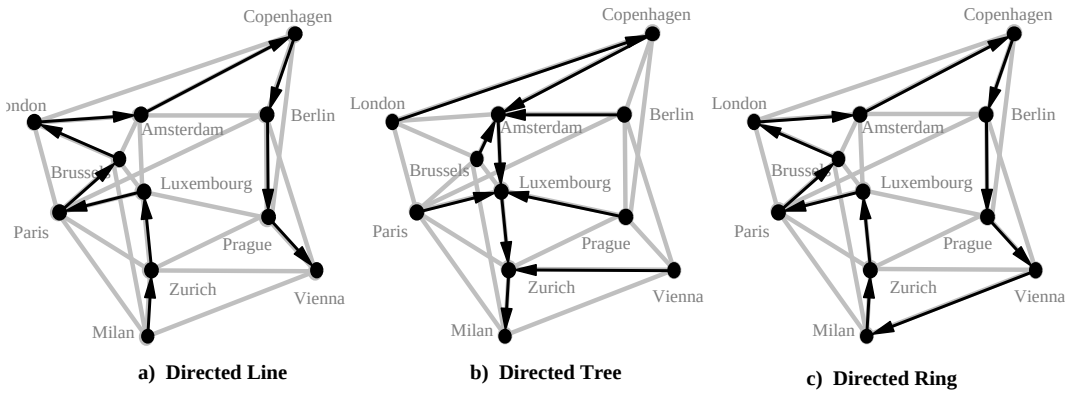


Figure 3.4: Forwarding Topologies

Examples of directed lines in existing networks include shortest paths and label switched paths (LSPs). Examples of directed trees include multipoint-to-point trees.

3.4 The Forwarding Table

A forwarding topology is established by providing at each node in that topology a record of the outgoing edge of that node used in that topology. The set of such records at a single node for all forwarding topologies that traverse that node is called the forwarding table.

A forwarding table entry records the following mapping:
forwarding topology → outgoing edge of the node.

3.5 Routing Assignments

When a packet enters the network (at the entry node), two decisions must be made in order to route the packet. The first is the choice of exit node from the nodes in that network.

A node must be chosen from which the destination address is reachable to be the exit node. If the network is multi-homed (i.e. possesses more than one gateway to external networks) there may be more than one choice of exit node. In such cases mechanism for selecting only one of the available exit nodes must be provided. Having chosen the exit node, a path from the entry node to the exit node must be selected. Using the abstraction of section 3.3, this corresponds to choosing a forwarding topology.

For example, in the scenario shown in Fig. 3.4 traffic from *Milan.** to *Vienna.** will follow forwarding topology *a* when destination addresses *Vienna.** are assigned to forwarding topology *a* at node *Milan*.

Traffic with destination address D, which enters network at node I, will be routed on forwarding topology F to exit node E, if a record is maintained at node I, listing D among the set of destination addresses reachable from node E in F. The process of mapping D to F at node I in this way is called routing assignment.

3.6 The Routing Table

The set of records of routing assignments at an entry node is called the routing table.

A routing table entry records the following mapping:
destination address, exit node $\rightarrow$ *forwarding topology*.

Prior to any routing assignment for a forwarding topology being made it must be confirmed that the destination address in that assignment is reachable from the exit node in that forwarding topology.

3.7 Routing Granularity

Different values can be mapped onto forwarding topologies in routing assignments. Three examples are:

- destination address prefixes,

- destination addresses, and
- source and destination addresses.

Routing granularity in these three cases is considered to be respectively coarse, base, and fine.

3.8 The Routing Algorithm

A routing algorithm is a set of rules which defines how destination addresses reachable from nodes in a forwarding topology are assigned to that forwarding topology, i.e. which defines how the routing assignments are made. Its output is the routing table.

3.9 Forwarding Layers

All routing assignments made for a single forwarding topology define a set of entry and exit nodes in that topology. That forwarding topology, and that set of entry and exit nodes define a forwarding layer. The options for the set of entry and exit nodes in a forwarding topology are:

- a single entry node and a single exit node,
- multiple entry nodes and a single exit node,
- a single entry node and multiple exit nodes, and
- multiple entry nodes and multiple exit nodes.

They correspond, respectively, to:

- point-to-point forwarding layers,
- multipoint-to-point forwarding layers,
- point-to-multipoint forwarding layers, and
- multipoint-to-multipoint forwarding layers.

Note that the existence of multiple exit nodes does not mean that a packet is delivered to multiple destinations. It means that some packets enter the forwarding topology at the same entry node, but they exit that topology at different exit nodes. Only unicast routing is considered here.

An example of each type of forwarding layer is given in Fig. 3.5. The three forwarding topologies shown in Fig. 3.4 are used. They are identified as F_a, F_b, and F_c, respectively. The shown routing assignments are provided for the marked entry node(s). Each assignment shows the destination addresses reachable from the exit nodes in the forwarding topology which are assigned to that forwarding topology.

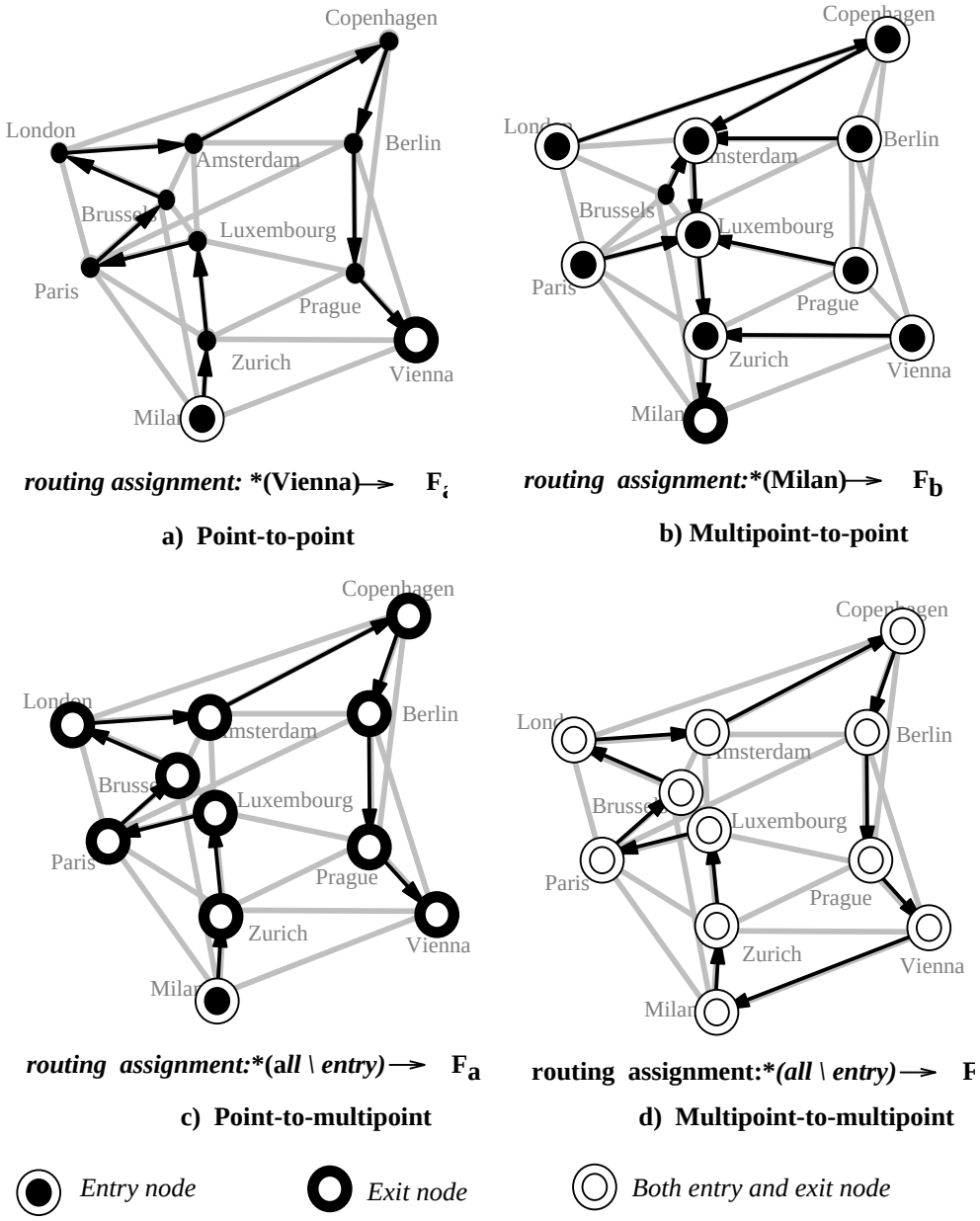


Figure 3.5: Forwarding Layers

Notation * *(exit node)* $\rightarrow$ *forwarding topology* implies that all destination addresses reachable from the exit node are assigned to the indicated forwarding topology.

Notation *(all \entry)* indicates all nodes except for the entry node for which the assignment is provided.

The routing algorithms used are as follows.

Point-to-point Routing Algorithm, Fig. 3.4a) (directed line): All addresses reachable from the root node are assigned to the forwarding topology at the leaf node.

Multipoint-to-point Routing Algorithm, Fig. 3.4b) (directed line or directed tree): All addresses reachable from the root node are assigned to the forwarding topology at every other node.

Point-to-multipoint Routing Algorithm, Fig. 3.4c) (directed line or directed tree): All addresses reachable from every node but the leaf node are assigned to the forwarding topology at the leaf node.

Multipoint-to-multipoint Routing Algorithm, Fig. 3.4d), (directed line, directed tree or directed ring): All addresses reachable from nodes which are at most μ hops away are assigned to the forwarding topology at every node.

Each described algorithm is applicable only for the type(s) of forwarding topologies specified in the brackets, which were defined in section 3.3.

A forwarding layer is a way of recording that the destination addresses reachable from the nodes in a forwarding topology are assigned to that forwarding topology by using the given routing algorithm.

3.10 Routing Topologies

Routing topologies are aggregates of forwarding topologies. For every routing topology there is an algorithm which splits that routing topology into a set of forwarding topologies. That algorithm is here called the splitting algorithm.

Three examples of routing topologies, called line, tree and ring, are shown in Fig. 3.6. Each topology has n nodes. The line in Fig. 3.6a) can be split into $n(n-1)$ directed lines by identifying the single paths between every two nodes. The tree in Fig. 3.6b) can be split into n directed trees by identifying, for each node, the set of single paths from all

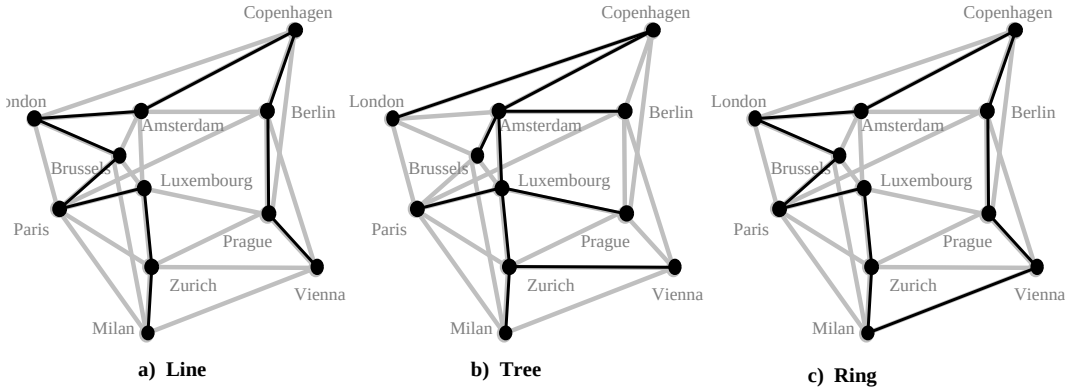


Figure 3.6: Routing Topologies

other nodes to that node. The ring in Fig. 3.6c) can be split into two directed rings by separating its edges into two sets depending on their direction at each node (*to* or *from*).

The network topology in Fig. 3.3 can also be a routing topology. Dijkstra's algorithm (section 2.2.2) determines a set of single shortest paths between every two nodes in a given network topology, and a set of shortest path trees, one for each node, i.e. this algorithm splits any given n-node network topology into $n(n-1)$ directed lines, and into n directed trees, provided that the connectivity between every two nodes exists.

3.11 Routing States

Routing assignments are made for a specified state of network and traffic. A routing assignment may become inapplicable when the state of a network element to which it routes traffic changes from active to inactive, i.e. when the network state changes. A set of routing assignments which route traffic to an active network element may become inadequate when traffic on that element increases above the level that can be handled by that element (*overload* occurs), i.e. when the traffic state changes. Failures to modify routing assignments when they become inapplicable or inadequate result in degradation of quality of service. Packets routed to an inactive network element are lost, so the quality of service for all traffic routed to an inactive network element during the time that element is inactive is severely degraded. The excess packets routed to an active network element during the time that the element is overloaded are dropped, and the remaining packets

may be delayed in queues. The quality of service may be degraded for all traffic sent to an overloaded network element during the time that element is overloaded.

A routing state is a set of network and traffic states for which a constant specified set of routing assignments applies.

The network state is defined by the states of network elements. Every edge and every node has two states: active and inactive. From the perspective of traffic a node being inactive is equivalent to all of the node's edges being inactive, so all network states can be expressed through the states of network edges only. When there are m network edges, there are in total 2^m possible network states.

3.12 Routing State Response Algorithm

Overall, a limited set of forwarding layers is used in the network. In a single routing state traffic follows a subset of forwarding layers in this set. In response to routing state changes the sets of destination addresses which are reachable in (and which will be assigned to) the forwarding topologies of these forwarding layers are modified by using different routing state response algorithms. A routing state response algorithm specifies the destination addresses reachable from the nodes in each forwarding topology in the current routing state. The reachable addresses are then assigned to forwarding topologies by the routing algorithm and the routing table is so created for the node where the routing algorithm is applied. This causes traffic shifts between paths in different forwarding topologies.

3.13 Routing Frequency

Updates of routing assignments as routing state changes ultimately causes traffic shifts across the network. Routing assignments can be updated whenever the routing state changes, or on particular transitions between two routing states only, e.g. when routing state changes from i to j, but not when it changes from j to i. The frequency of updates of routing assignments is the routing frequency.

The routing frequency impacts the quality of service and the efficiency in using network resources. Longer delays in updating routing assignments may cause a higher packet

loss. Instant updates of routing assignments which cause traffic from a long path through the network to be shifted to a short path may cause out of order delivery of information which is sent in a stream of packets. Frequent shifts of traffic between a number of paths which form a loop may cause out of order and delayed delivery of information sent in a stream of packets, and waste of network resources, as some packets may end up looping through the network for a long time. Frequent updates of routing assignments which require network protocol information to be exchanged between network nodes increase waste of network resources on network protocol traffic.

3.14 Participants and Responsibilities

The participants in executing tasks in a routing solution are:

- the network provider, and
- the network protocols, which are:
  - the forwarding protocol,
  - the routing protocol, and
  - the control protocol.

The responsibilities are divided between the participants as follows.

The network provider plans, establishes, maintains and modifies the network, and provides the configuration information.

The forwarding protocol forwards packets in the order of their priority by following the forwarding instructions.

The routing protocol monitors routing states, and provides the forwarding instructions for the forwarding protocol in the current routing state at each network node. The forwarding and routing tables are created first by applying algorithms at each node which process the configuration information provided and the monitored routing state information. The information in the forwarding and routing tables is subsequently used for creating the forwarding instructions in the current routing state at that node.

The control protocol distributes configuration information from the network control centre to all network nodes, and it gathers routing state data for the network control centre.

The forwarding protocol and the routing protocol operate in a distributed manner, meaning that they perform an identical set of operations at each network node. The control protocol operates in a centralized manner. It distributes requests from a single node, where the network control centre accesses the network, and gathers feedback to that node.

3.15 Traffic Control

Traffic routes are selected:

- by planning the network, and
- by choosing a routing strategy, i.e. by defining:
  - the routing states, and the routing frequency,
  - the forwarding topologies in every routing state, possibly by defining routing topologies in every routing state, and
  - the routing assignments in every routing state, or the routing algorithm and the routing state response algorithm.

The listed decisions ultimately drive and control network traffic. They are realized through various traffic control mechanisms.

The routing strategy is chosen by the routing authority, which is a body which defines standards and policies, and/or the network provider. It is imposed by formulating:

- the configuration information, to be provided at each node directly, and remotely, from the network control centre,
- the network protocols, and
- the algorithms to be deployed by the network protocols.

The network users can be allowed to use the network according to their preferences. Users express their preferences by specifying demands. A demand states the traffic source and destination address, the level of quality of service requested, and the amount of network resources required. The routing authority defines rules for mapping a demand into a routing assignment in the current routing state for the destination specified in the demand, given the current state of network resources. The user requests are queued and served until network resources are exhausted. The traffic specified in every approved demand is marked (prioritized) according to the quality of service requested in that demand. If more traffic ends up on a network element than that network element can handle, i.e. if a network element becomes overloaded, the quality of service is degraded for the low(er) priority traffic on that network element during the time that element is overloaded. The high(er) priority traffic which is within the traffic level that can be handled is not affected.

3.16 Protocol Specifications

The procedures which have to be specified in network protocol specifications are:

- the forwarding procedure, and
- the procedure for providing forwarding instructions.

The set of operations required in providing forwarding instructions for a single routing state includes:

- establishment of forwarding tables,
- establishment of routing tables, and
- establishment of tables with forwarding instructions.

Providing forwarding instructions for a set of routing states requires:

- monitoring of routing states, i.e.
  - monitoring of network states, and

  - monitoring of traffic states,

- update of forwarding tables,
- update of routing tables, and
- update of tables with forwarding instructions.

The operations required for allowing network users to use the network according to their preferences are:

- mapping of demands into routing table entries,
- management of network resources, and
- prioritization of packets, which includes
  - providing instruction for marking of packets, and
  - marking of packets according to their priorities.

The last operation can be accomplished on the network protocol layer, or on the layer above in the layered protocol architecture.

3.17 Network Planning and Maintenance

The minimum information that the network provider needs in planning the network is a rough estimate of:

- the number and locations of destinations that are expected to exchange traffic across the network, and
- the expected traffic demands between the destinations.

If the network provider is not the routing authority, the routing authority also has to specify the planned paths in every routing state.

The network provider defines the network so that there are sufficient resources for η times the expected traffic on the planned paths in the most likely network states, where $\eta \geq 1$ is the over-provisioning factor. Examples can be found in [103, 104, 105].

The routing state depends largely on the network provider.

Changes in network state are rare, and result from unexpected accidents and pre-planned maintenance. It is the responsibility of the network provider to ensure that any maintenance is carried out quickly. With that condition fulfilled, the likelihood for the network to have k inactive edges decreases as k increases. It is unlikely for the network to have $k > \nu$ inactive edges. Provided that traffic estimates were *sufficiently* accurate, traffic on every planned path between every two nodes is likely to be below the level of expected traffic on that path in the initial *long* time interval. As new users join the network and new opportunities for the network service application arise, the demands on the planned paths increase gradually over time. It is the responsibility of the network provider to provide gradually more network resources, according to the newly estimated requirements. With that condition fulfilled, traffic on the planned paths is likely to be maintained within the expected boundaries.

Given the responsibilities the network provider has, its performance greatly depends on the speed and quality of the feedback that is required for these responsibilities to be carried out. For ensuring that maintenance of network elements is carried out fast an immediate feedback on the network state changes needs to be provided. For ensuring that more resources are provided in the network in a timely fashion as need arises a *slow* feedback on traffic states on the planned paths is needed, e.g. on a daily basis. As network planning has to be based on traffic estimates which might turn out to be *significantly* wrong, the network provider should be able to modify the initially planned traffic routes. Also, for minimizing the risk taken in deploying a routing strategy, options should be provided for the network provider to modify the routing strategy relatively simply and fast.

In summary, the performance of the network provider greatly depends on the existence of mechanisms for fast and efficient central control of traffic routes, as well as for central control of the overall network operation. A control protocol is necessary which would distribute commands and requests to all network nodes from a single location, i.e. from the network control centre, and gather feedback to that single location.

3.18 The Forwarding Protocol

The role of the forwarding protocol in packet-switched networks corresponds to the role of professional drivers on the roads. The forwarding protocol guides each packet from one destination to another. For its operation this protocol requires knowledge of:

- the table(s) with forwarding instructions at each node, and
- the forwarding header in each packet.

3.18.1 Tables with Forwarding Instructions

A forwarding instruction at a node specifies the outgoing edge of that node where each packet with specified information in its forwarding header has to be forwarded. The instruction may also specify which information has to be recorded to or removed from the forwarding header of the packet prior to that packet is forwarded.

A packet which enters the network has to be sent along a forwarding topology from the entry node to the exit node for that packet in that forwarding topology. At subsequent nodes in respect to the entry node in the forwarding topology that the packet follows the packet has to either exit the forwarding topology, or continue in the direction of that forwarding topology. Upon exiting the forwarding topology the packet either begins to follow another forwarding topology, or exits the network. Overall, the packets which arrive at a node may:

- enter a forwarding topology at that node (after entering the network or exiting another forwarding topology),
- exit a forwarding topology at that node, or
- continue down the same forwarding topology (i.e. neither enter nor exit a forwarding topology) at that node.

These packets are called respectively the incoming packets, the outgoing packets, and the transit packets. The same packet forwarded through the network is the incoming packet at the entry node, the transit packet on its subsequent hops, and the outgoing packet at the

exit node. The incoming and the outgoing packets are also referred to in this thesis as the non-transit packets.

3.18.2 The Forwarding Header

In the forwarding header of a packet the forwarding protocol records the information that it needs along the way to deliver that packet to its destination. Which information is needed depends on how the forwarding instructions are formulated. At the start of the packet's journey, the forwarding protocol records the relevant information available at the packet's source. At the entry node to a forwarding topology, and possibly at the other nodes in that topology, the forwarding protocol may record some information that it needs for guiding that packet down that forwarding topology which is obtained at these nodes. For every item of information that needs to be recorded, a field has to be provided in the forwarding header. The location and the processing of the fields for the information that the nodes in a forwarding topology provide may differ. Three examples follow.

- These fields are reserved in the forwarding header at the source of the packet, at an arbitrary position in the header, as shown in Fig. 3.7b).
- The forwarding header provided at the source of the packet is extended at the end by a set of fields when the packet reaches a forwarding topology, at the entry node, as shown in Fig. 3.7c). These fields are removed before the packet exits the forwarding topology, at the exit node.
- The forwarding header provided at the source of the packet is extended in front by a set of fields when the packet reaches a forwarding topology, at the entry node, as shown in Fig. 3.7d). These fields are removed at the exit node.

Two options where the information that the forwarding header has been extended, when that is the case, can be carried in the layered protocol architecture are:

- on the network protocol layer, or
- on the layer below the network protocol layer.

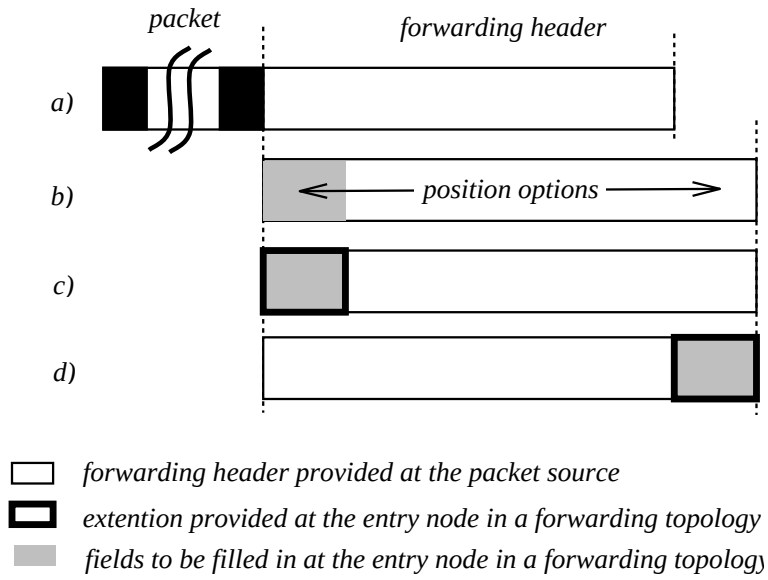


Figure 3.7: Forwarding Header

The choice between these two options affects the speed of the forwarding process.

Suppose that there is a set of *basic* fields in the forwarding header of a packet at the packet's source, and that *additional* fields (defined either as a general set or as a standardized set of fields) are added to this header when the packet reaches a forwarding topology, at the entry node, to be removed from the header before the packet exits the forwarding topology, at the exit node. The information that these additional fields exist is of relevance to the forwarding protocol, and, if one is to comply with the layered protocol architecture, it should be carried on the network protocol layer where this protocol resides. This can be achieved by recording in one of the basic fields in the forwarding header that a set of additional fields had been placed **behind** the basic fields at the entry node. In that case, for the additional fields to be examined at subsequent hops, their existence has to be confirmed first by examining the basic fields, i.e. both the basic and the additional fields have to be examined for the information in the additional fields to be retrieved. The examination of the additional fields can be sped up by placing the additional fields **in front** of the basic fields, and by sending the information that such a placement of fields had been performed in the forwarding header on the layer below the network layer in the layered protocol architecture, i.e. by breaking the layered protocol architecture. With the additional fields placed first, the basic fields can be ignored when only the information from the additional fields is needed. So, the described manipulations can speed up the

header examination in the forwarding process for transit packets.

Handling of a packet that belongs to one protocol by another protocol which for its operation uses a separate header placed in front of the packet is called *encapsulation* or *tunnelling*. The additional fields placed in front of the basic fields in the forwarding header can be viewed as a separate packet header, and so the forwarding protocol which operates with that way organized forwarding header can be considered a *tunnelling mechanism*. A forwarding protocol which operates with the alternatively organized forwarding header is not a tunnelling mechanism. This does not imply that its forwarding speed is lower. That depends on the duration of all other operations involved in the forwarding process. The benefit of *tunnelling* is in speeding up the forwarding header processing.

3.19 Summary

A high level summary of the operations involved in routing of packets in packet-switched networks in general, including a common framework that existing routing solutions follow, as seen by the author of this thesis, has been presented in this chapter. The aim of this chapter was primarily to establish grounds for a more detailed analysis, and to present the terminology used in succeeding chapters.

Chapter 4

A Comparative Analysis of IP and MPLS

Two of the intra-area routing solutions described in chapter 2 are described in this chapter using the common basic framework presented in chapter 3. The focus in the analysis is on the forwarding protocol, routing strategy, and the establishment and update of forwarding and routing tables. The major differences between these routing solutions are subsequently summarized in the chapter.

4.1 The Forwarding Protocol

As discussed in section 3.18, the forwarding protocol requires for its operations:

- the table(s) with forwarding instructions at each node, and
- the forwarding header in each packet.

At each node there are up to three streams of packets for which forwarding decisions have to be made. They are the incoming packets, the outgoing packets, and the transit packets at that node, as defined in section 3.18.1.

The choices made in forwarding protocols IP and MPLS are examined in this section.

4.1.1 IP

In the IP forwarding protocol used in hop-by-hop IP routing the following restriction is imposed on the routing strategy.

A destination address reachable from an exit node which is assigned to a forwarding topology at an entry node must be assigned to the same forwarding topology at every node on the path from the entry node to the exit node in that forwarding topology.

This restriction ensures that the incoming and transit packets which go to a single destination merge onto a single forwarding topology at each node.

The forwarding protocol is then provided with:

- a field in the forwarding header for recording the source and destination address of the packet at the packet's source, and
- a table with forwarding instructions at each node for both transit and non-transit packets at that node.

A forwarding instruction in this table contains the following mappings:
destination address prefix $\rightarrow$ *outgoing edge of the node*.

It indicates the outgoing edge of the node where each packet with the destination address that matches the specified destination prefix has to be forwarded.

The instructions are made based on the information in the routing and forwarding tables, and based on the destination addresses known as reachable from each egress edge of the node. The instructions for the outgoing packets which exit the network at the node are a copy of the destination addresses reachable from each egress edge of that node. The routing table gives the forwarding topology which each incoming packet with the specified destination address has to follow. The forwarding table gives for each forwarding topology the outgoing edge of the node where the packet has to be forwarded. Provided that the specified restrictions are obeyed, the forwarding instructions for the incoming packets apply for the transit packets too.

The forwarding header for the IP forwarding protocol is organized as follows. There

is a standardized set of b basic fields of a fixed length, ℓ_b. Any set of additional fields of variable length, ℓ_a, can be added to the forwarding header as the packet traverses the network, and be placed behind the standardized set of b basic fields. The total length of the forwarding header, $\ell_a + \ell_b$, must then be recorded in one of the basic fields.

Note: There is only one table with forwarding instructions in IP and it is called the routing table. The forwarding header is called the IP header.

4.1.2 MPLS

In MPLS forwarding protocol used in explicit MPLS routing there is a separate table with forwarding instructions for incoming, for outgoing, and for transit packets.

For forwarding an incoming packet this forwarding protocol requires:

- a field in the forwarding header for recording the source and destination address of the packet at the packet's source, and
- a table with forwarding instructions at each node for incoming packets at that node.

There is flexibility in the choice of routing granularity in routing assignments. A forwarding instruction in the table with forwarding instructions for incoming packets may contain the following mappings:

destination address prefix, outgoing label $\rightarrow$ *outgoing edge of the node*, or
destination address, outgoing label $\rightarrow$ *outgoing edge of the node*, or
source and destination address, outgoing label $\rightarrow$ *outgoing edge of the node*.

It indicates the outgoing edge of the node where each packet either with the destination address that matches the specified destination prefix, or with the specified destination address, or with the specified source and destination address, has to be forwarded. The instructions are made based on the information in the routing and forwarding tables, and the destination addresses known to be reachable from each egress edge of that node. The

outgoing labels are identifiers specified in the process of *label binding* and *label distribution*. The forwarding header of the packet has to be prefixed with a set of fields at the entry node, and the corresponding outgoing label has to be recorded in the added fields before the packet is forwarded.

Let a *relevant routing assignment* for node N_i be any routing assignment made at other network nodes $N_{j,j\neq i}$ which causes packets to traverse node N_i. The *corresponding forwarding topology* is the forwarding topology in that relevant routing assignment. For the labels in the tables with forwarding instructions to be provided, the following restriction is imposed on the procedure for providing forwarding instructions.

A label *has to be assigned independently for each node N_i to every relevant routing assignment for node N_i. That label also has to be provided at the predecessor node(s) in the corresponding forwarding topology as an* outgoing label.

This operation of *label binding* and *label distribution* maps the routing table entries into one or two labels. At each node there is an *outgoing label* for each routing table entry at that node. There are two labels for each routing table entry of other nodes which contain relevant routing assignments for that node: the *label*, and the *outgoing label*.

The forwarding protocol is then provided with:

- a field in the forwarding header for recording at each node in the forwarding topology that the packet follows the outgoing label, and
- a separate table with forwarding instructions at each node for transit packets at that node.

A forwarding instruction in this table contains the mappings:
label, outgoing label $\rightarrow$ *outgoing edge of the node.*

It indicates the outgoing edge of the node to which each packet with the specified label has to be forwarded. The provided outgoing label has to be recorded instead of the label in the forwarding header of that packet. This forwarding instructions are made based on the labels and the outgoing labels which were assigned during label binding and distribution

to the relevant routing assignments, and based on the forwarding table at the node which has records of the node's outgoing edges in corresponding forwarding topologies.

The existence of the additional fields in the forwarding header of a packet, which were provided at the entry node, identifies that packet as a transit or an outgoing packet at subsequent nodes in the forwarding topology that the packet follows. A packet with the extended forwarding header is a transit packet if for the label that it carries there is an outgoing label in the table with forwarding instructions for transit packets. Otherwise, the packet is recognized to be an outgoing packet. The forwarding instruction for the packet is then retrieved from the table with forwarding instructions for outgoing packets.

A forwarding instruction in the latter table contains the following mappings:

destination address prefix $\rightarrow$ *outgoing edge of the node*.

It indicates the outgoing edge of the node to which each packet with the destination address that matches the specified destination prefix has to be forwarded. The additional fields which were added to the forwarding header of the packet at the entry node have to be removed before the packet is forwarded.

The forwarding header for the MPLS forwarding protocol is organized as follows. There is a standardized set of b basic fields of a fixed length. A variable number κ of identical standardized sets of a additional fields, each of a fixed length, l_a, are added in front of this basic set of fields as the packet enters the network. A one-bit flag in the first set of a additional fields is reserved for recording whether κ equals one or not, i.e. whether the total length of the additional fields is no more than l_a. This flag has to be updated accordingly if the processing of the additional fields results in a different value of κ.

Note: The set of basic fields of the forwarding header is that used in IP and it is called the IP header. A single set of a additional fields is called the MPLS header. The MPLS forwarding protocol can be regarded as a tunnelling mechanism. The table with

forwarding instructions for outgoing packets in the MPLS forwarding protocol is the IP routing table . The remaining tables with forwarding instructions are called the forwarding tables.

4.1.3 Comparison of the Protocols

The major novelties introduced in the MPLS forwarding protocol when compared to the IP forwarding protocol are in:

- the forwarding header processing,
- the identification of next hops in tables with forwarding instructions, and
- the label binding and label distribution schemes.

MPLS has traded packet header for packet processing. Within the four bytes it adds to the forwarding header of each packet this protocol records a short identifier, called a label (which is associated to the routing assignment that guides forwarding of that packet) thus increasing the forwarding header overhead. The benefit of this is in speeding up processing of the forwarding header compared to IP where a header twenty bytes long must be examined (see also section 3.18.2). The newly introduced labels are used in MPLS for the identification of next hops in tables with forwarding instructions whereas for the same purpose address prefixes are used in IP. The benefit of the former solution is that an exact match table lookup can be performed in the forwarding process, which is faster than the longest prefix matching lookup required in the latter case. However, by having the labels defined in MPLS independently for each node so that the label a packet carries must be changed at each hop, as opposed to having them defined so that the label is identical while the packet traverses a particular forwarding topology (or area), hop-by-hop header modifications are introduced in the forwarding process in MPLS which are not necessary in IP. Also a need for various label management schemes has been created. Whether the benefits of having the labels locally defined justify this overhead is questionable.

Due to the choice of forwarding instructions for the forwarding protocol and the definition of the forwarding header in IP and MPLS, these two protocols differ in:

Table 4.1: IP vs. MPLS

	IP	MPLS
Forwarding header overhead:	$\underline{s}+\underline{d}$	$\underline{s}+\underline{d}+\underline{r_N^*}$
Forwarding instructions overhead:	x	$x+r_N^*$
Overhead in Providing Instructions:	none	some
Limitations on Routing Strategy:	some	none
Forwarding delay, incoming packets:	t_{lpl}	$t_{lpl}+t_{hm}$
Forwarding delay, transit packets:	t_{lpl}	t_l+t_{hm}
Forwarding delay, outgoing packets:	t_{lpl}	$t_l+t_{hm}+t_{lpl}$

- the overhead of the forwarding header,
- the forwarding instruction overhead at each node,
- the overhead in providing forwarding instructions,
- the limitations imposed on the routing strategy, and
- the speed of the forwarding process.

The parameters affecting the performance of the two forwarding protocols are shown in Table 4.1, where s is the source address, d is the destination addresses, r_N^* is the number of relevant routing assignments for a given node, $\underline{s}$, $\underline{d}$, $\underline{r_N^*}$ are the lengths of their identifiers, respectively, x is the overhead of forwarding instructions for non-transit packets, t_{lpl} is the time required for a table lookup using longest prefix matching, t_{hm} is the time required for modification of a field in the forwarding header, and t_l is the duration of a simple table lookup, which is faster than the longest prefix lookup (i.e. ,$t_l < t_{lpl}$).

The choices made in IP have led to a lower overall overhead, but limitations on the routing strategy are introduced. MPLS avoids these limitations on the routing strategy, but the overall overhead introduced to achieve this is significant. The label management which MPLS requires in the process of providing forwarding instructions is complex. It increases network protocol message overhead, it introduces additional header manipulations in the forwarding process, and its implementation depends on the type of forwarding topologies used. The forwarding speed in the two cases depends primarily on the speed of lookups of the tables with forwarding instructions and on the speed of network links.

4.2 Routing Strategies

As discussed in section 3.15, the routing strategy is specified by choosing:

- the routing states, and the routing frequency,
- the forwarding topologies in every routing state, possibly by defining routing topologies in every routing state, and
- the routing assignments in every routing state, or the routing algorithm and the routing state response algorithm.

The initially made choices of routing strategies in hop-by-hop IP routing and explicit MPLS routing are examined in this section.

4.2.1 Shortest Path Routing and a Virtual Network Model of Link Costs

So-called shortest path routing is the routing strategy most commonly used in hop-by-hop IP routing. Traffic is in this case directed to the shortest paths through the network. That is, at each node, all destination addresses reachable from a node are assigned to the shortest path through the network to that node. The shortest paths to a node from every other node form a directed tree or a directed line forwarding topology. All destination addresses reachable from its root node are assigned to that forwarding topology at every other node. So, there are n multipoint-to-point forwarding layers that traffic follows, where n is the number of nodes in the network. The length or cost of each edge in physical network topology is unity.

Routing state changes are modelled using virtual networks. The virtual network topology contains the same network elements as the physical network topology, but the lengths of network edges in these two topologies differ. An edge in a virtual network topology can have 2^{χ} different lengths, where $\chi = 16$ if link costs are recorded in two bytes. There are $(2^{\chi})^m$ possible virtual networks that can be derived from the physical network, where m is the number of edges.

When the routing state is fixed all traffic is assigned to a single virtual network. The selection of traffic routes in a single routing state requires:

- the sets of destination addresses which are reachable from each node in the network,
- one of the $(2^{\chi})^m$ virtual network topologies to be used as the routing topology,
- the Dijkstra's algorithm as the splitting algorithm (section 3.10), and
- the multipoint-to-point routing algorithm (section 3.9).

Traffic is thus directed to up to n virtual correlated multipoint-to-point forwarding layers. Each virtual multipoint-to-point forwarding layer maps into a physical multipoint-to-point forwarding layer in the physical network which traffic actually follows.

In response to changes in routing state traffic is switched to a different virtual network, which may cause traffic in the physical network to be shifted from one set of multipoint-to-point forwarding layers to another. Many of the available virtual topologies will result in the same outcome when Dijkstra's algorithm is applied. Thus a change in a routing state, and thus a migration to a new virtual topology, may not result in traffic shifts in the physical network. Inactive edges of the physical network are mapped to edges of maximum length in the corresponding virtual topology.

4.2.2 Constraint-Based Routing

So-called constraint-based routing was one of the first routing strategies considered in explicit MPLS routing. In this case traffic between a source and a destination is directed to a path through the network which fulfils a set of given constraints, as follows.

If d destination addresses are reachable from the nodes in each of the $(2^{\chi})^m$ virtual networks used in shortest path routing (section 4.2.1), each of these virtual networks is expanded into d virtual networks which have only a single destination reachable from each exit node. The number of available virtual networks in thus increased to $(2^{\chi})^m \times d$.

The increment of traffic between two destinations can be predictable or unpredictable. Predictable traffic increments are announced by network users in their demands. Traffic is assigned to a virtual network when a traffic increment is guaranteed by a network user,

i.e. when a demand arrives from a network user for network service of a particular quality. The selection of the route for this traffic requires:

- the single destination address in that demand which is reachable from a node in the $(2^{\chi})^m$ virtual networks,
- the selection of a virtual network topology as the routing topology,
- Dijkstra's algorithm as the splitting algorithm, and
- the point-to-point routing algorithm (section 3.9).

4.2.3 Multiple Topology Routing

So-called multi-topology routing extensions have recently been considered in IP networks. In this case all traffic is assigned to a set of up to 2^{φ} virtual networks in a single routing state, out of the $(2^{\chi})^m$ virtual networks available in shortest path routing, where typically $\varphi = 8$ and $\chi = 16$. These 2^{φ} virtual networks have different topologies. Hence traffic is directed to up to $2^{\varphi} \times n$ virtual multipoint-to-point forwarding layers.

An alternative to the constraint-based routing in MPLS networks is to use multiple forwarding layers concurrently within a single routing state. These are defined centrally. The minimum number of point-to-point forwarding layers in such a set is $n(n-1)$. Traffic on a forwarding layer affected by a change in network state is switched to a pre-defined backup forwarding layer.

4.2.4 Comparison of Routing Strategies

A major difference between the described routing strategies is in the type of forwarding layer used. For providing connectivity between nodes in an n-node network a single routing topology is sufficient in shortest path routing, and $n(n-1)$ directed line forwarding topologies are needed in constraint-based routing, and so a substantially lower number of topologies is needed in the former case. As the routing granularity in shortest path routing is coarse, and in constraint-based routing is fine, the number of routing assignments is substantially lower in the former case, too. But, only with point-to-point forwarding

layers, used in the latter case, is it possible to modify routing assignments independently at arbitrary network nodes and cause traffic shifts from one route to another without disrupting the remaining network traffic. Such actions are not possible when only a single routing topology is used. Traffic control options are thus fairly limited in the latter case.

The minimum number of forwarding layers of a single type which is required for providing connectivity between every two nodes in a single routing state in an n-node network is:

- n(n-1) point-to-point forwarding layers,
- n point-to-multipoint forwarding layers,
- n multipoint-to-point forwarding layers, and
- 1 multipoint-to-multipoint forwarding layer.

At least double this number of layers is needed if connectivity is to be maintained whenever any single edge becomes inactive.

A single ring topology which connects all network nodes aggregates two multipoint-to-multipoint forwarding layers which are sufficient for providing connectivity between all nodes, and for ensuring that the connectivity exists when any single edge becomes inactive. But, it is likely that some traffic would then have to take an excessively large number of hops to traverse the network. A single routing topology which connects all network nodes may aggregate up to:

- n(n-1) point-to-point forwarding layers, or
- n point-to-multipoint forwarding layers, or
- n multipoint-to-point forwarding layers.

These correlated forwarding layers are sufficient for providing connectivity between all nodes. But, additional forwarding layers are needed for ensuring that connectivity is maintained when any single edge becomes inactive.

4.3 Providing Forwarding Instructions

The overhead incurred in providing forwarding instructions in IP and MPLS networks will now be considered.

4.3.1 Hop-by-hop IP Routing

The establishment of tables with forwarding instructions in hop-by-hop IP routing requires the following operations.

Establishment of forwarding tables: Forwarding topologies are established, i.e. forwarding table entries are created, by specifying for each node the set of edges of that node in a routing topology. Subsequently:

- each node periodically advertises its edges in each routing topology in the network to all other network nodes,
- the routing topologies are reconstructed at each node based on the advertisements received from all other nodes,
- the routing topologies are split into forwarding topologies at each node, and
- the forwarding table entries are created for each node.

All relevant information is distributed to all network nodes by being flooded across all network edges.

Update of forwarding tables: Forwarding topologies are established periodically. The periodic establishment of forwarding topologies introduces delay which may be critically prolonged when network state changes are frequent and the establishment of forwarding topologies is restarted a number of times. During that time traffic on routes that have become invalid due to network state changes is randomly routed through the network. This leads to an unnecessary waste of network resources and to an out of order and delayed delivery of information sent in a stream of packets, i.e. to a degraded quality of service.

Monitoring of network states: Network state changes are monitored by monitoring the states of network nodes and edges, which is accomplished by sending periodic test messages along each network edge.

Establishment and update of routing tables: Once the destination addresses reachable from each network node are specified, network nodes periodically advertise the destination addresses reachable from themselves. Based on that information and the known forwarding topologies, routing assignments are periodically made at each node by applying the multipoint-to-point routing algorithm.

4.3.2 Explicit MPLS Routing

The establishment of tables with forwarding instructions in explicit MPLS routing requires the following operations.

Establishment of forwarding tables: Forwarding topologies are established, i.e. forwarding table entries are created, by specifying for a single node the forwarding topologies that contain that node. At each node the forwarding table entries are then created for that node.

Information is distributed to network nodes in a network protocol message together with an ordered list of nodes which specifies the sequence of nodes to receive that information. Each node forwards that message to its successor in the sequence of nodes, and confirms the receipt of the information to its predecessor in the sequence.

Update of forwarding tables: Whenever a forwarding topology becomes inapplicable a new forwarding topology is established to replace that forwarding topology. The establishment of forwarding topologies introduces delay which may be critically prolonged when network state changes are frequent and the establishment of forwarding topologies is restarted a number of times. During that time traffic on inapplicable routes is dropped which degrades quality of service.

Monitoring of network states: Changes of network states affect forwarding topologies and routing assignments made in the affected forwarding topologies. Network state changes are monitored either:

- by monitoring the states of routing assignments and forwarding topologies, which is accomplished by sending periodic test messages upstream in each forwarding topology in current use (opposite the direction of its edges) for each routing assignment made in that topology, or
- by monitoring the states of network nodes and edges, which is done by sending periodic test messages along each network edge, and then by sending upstream notifications along each affected forwarding topology for each routing assignment made in that topology whenever state changes of network elements in that topology are detected.

Establishment and update of routing tables: Once the destination addresses reachable from each network node are specified, network nodes advertise destination addresses reachable from themselves network wide periodically. Based on that information and the known forwarding topologies, routing assignments are made at each node by applying the point-to-point routing algorithm. Whenever the state of a routing assignment changes, or sets of reachable destination addresses change, the routing table entry for that routing assignment is updated.

Label binding and distribution: The process of label binding and distribution, described in section 4.1.2, is performed whenever routing tables are updated. In this process the labels associated with routing assignments are recorded in the corresponding tables with forwarding instructions.

4.3.3 Comparison of Overheads

The operations that precede the establishment of tables with forwarding instructions differ significantly in hop-by-hop IP routing and explicit MPLS routing. These differences are primarily a consequence of the choice of forwarding layers in the two cases, which leads to a considerable difference in the number of topologies required to establish forwarding tables in the two cases. In hop-by-hop IP routing a single routing topology which aggregates n directed tree forwarding topologies is sufficient to specify the forwarding tables for a single routing state in an n-node network, whereas in explicit MPLS routing

at least $n(n-1)$ directed line forwarding topologies are needed, where n is the number of network nodes. Another important difference is in routing granularity, which is coarse in hop-by-hop IP routing, and fine in explicit MPLS routing. So the number of routing assignments in the latter case is much higher. Consequently, there is a substantial difference in the information overhead, processing overhead, and message overhead in the two cases. Significantly less protocol messages are required for monitoring network states in hop-by-hop IP routing than in explicit MPLS routing, and the forwarding tables, routing tables and tables with forwarding instructions are significantly smaller in size. Extra complexity and message overhead is incurred by the label binding and label distribution needed in explicit MPLS routing. In both routing solutions the delay in establishing forwarding topologies in response to changes in the current routing state leads to a degradation of quality of service, and proposals have been made to minimize such negative effects by diverting traffic to pre-established backup forwarding topologies following network state changes (sections 2.2.7 and 2.4.6). The proposed solutions for the establishment of such backup forwarding topologies in the two cases differ significantly.

4.4 Improving Routing Protocols

A comparative analysis of two existing intra-area routing solutions has been conducted in this chapter. The analysis has shown the following.

Hop-by-hop IP routing has traded off flexibility in traffic control against simplicity in network management and data overhead, which is achieved by using only one routing topology and coarse routing granularity. This solution offers very limited traffic control options. Explicit MPLS routing requires complex network management and considerable data overhead, as it supports *any* number of directed line forwarding topologies and fine routing granularity. This allows much flexibility in traffic control, but there is scope for improvement. In both solutions there are types of forwarding layers which cannot be used. Options are lacking for the establishment of all types of forwarding topologies and the use of routing topologies in this process, and packet forwarding cannot be performed along any type of forwarding topology. While the presence of point-to-point forwarding layers is important for providing flexibility in traffic control, the presence of other types

of forwarding layers and the use of routing topologies are important for reducing the information overhead and for simplifying the network management.

Many improvements for both solutions have been proposed in recent years. In hop-by-hop IP routing, in order to speed up update of forwarding tables and for the traffic control options to be increased, it has been proposed to start using a number of routing topologies in a single routing state instead of only a single routing topology (sections 2.2.7 and 2.2.10). In explicit MPLS routing changes were considered so that the update of forwarding tables is sped up by diverting traffic to backup pre-established forwarding topologies following network state changes (section 2.4.6). Additional modifications were proposed in order for the information overhead and complexity of network management to be reduced, and for the scalability issues to be avoided (section 2.4.7).

In the reminder of this thesis a routing solution is proposed which allows fine traffic control with moderate overhead.

4.5 Design Principles for a New Routing Protocol

Many different routing strategies for packet-switched networks can be defined. The network provider may use simulations to choose among these. Such simulations are extremely complex given that, for example, the number of probable routing states is excessive, and traffic state reports and estimates used in simulations may be inaccurate. To minimize the risk the network provider takes in deploying a routing strategy, it should be possible for any chosen routing strategy to be readily modified. The routing solutions considered here do not provide such flexibility. To achieve this, mechanisms have to be provided for modifying in a centralized manner (in a control protocol) factors that affect traffic routes, and a forwarding protocol compatible with any type of forwarding topologies is needed. Such protocols are proposed in the next chapter.

Chapter 5

New Routing Solutions for IP Networks

The need was identified in the previous chapter for new network protocols and new routing strategies so as to provide flexible traffic control with modest overhead. A new forwarding protocol, a new control protocol, and new routing strategies which make use of them are described below. Also described are two new routing topology algorithms for traffic control purposes in networks which deploy the described protocols.

5.1 The Forwarding Protocol

This is based on the MPLS forwarding protocol (section 4.1.2) so as to retain its advantage of fast header processing. As in MPLS, each packet carries a short identifier in the field prefix of its forwarding header, which is recorded there at the entry node of a forwarding topology and which guides that packet down that forwarding topology. But, unlike in MPLS, the short identifier which a packet carries does not change as the packet hops down the forwarding topology. It is a unique identifier for a particular forwarding topology. The uniqueness of the forwarding topology identifiers is assured by imposing a set of rules. This avoids the complex label binding and distribution schemes of MPLS that inhibit its capacity to support novel routing strategies if their implementations require new label binding and distribution schemes. A more detailed description of the proposed forwarding protocol follows.

The forwarding protocol treats differently the incoming packets, the outgoing packets,

and the transit packets (as defined in section 3.18.1). It requires two tables with forwarding instructions, one for non-transit packets and one for transit packets.

For forwarding non-transit packets this forwarding protocol is provided with:

- a field in the forwarding header for recording the source and destination address of the packet at the packet's source, and
- a table with forwarding instructions at each node for non-transit packets at that node.

For forwarding transit packets the forwarding protocol is provided with:

- two fields in the forwarding header for recording at the entry node to a forwarding topology:
  - the identifier of the forwarding topology that the packet follows, and
  - the identifier of the exit node for that packet in that topology, and
- a separate table with forwarding instructions at each node for the transit packets at that node.

The forwarding protocol allows flexibility in the choice of routing granularity in routing assignments. A forwarding instruction in the table with forwarding instructions for non-transit packets may contain the following mappings:
destination address prefix, exit node, forwarding topology → outgoing edge of the node
destination address, exit node, forwarding topology → outgoing edge of the node
source and destination address, exit node, forwarding tpl → outgoing edge of the node

It indicates the outgoing edge of the node where each packet either with the destination address that matches the specified destination prefix, or with the specified destination address, or with the specified source and destination address, has to be forwarded. The instructions are made based on the information in the routing and forwarding tables, and

on the destination addresses specified as reachable from that node. The forwarding header of the packet has to be prefixed with a set of fields at the entry node. The corresponding identifier of the forwarding topology which the packet follows and the identifier of the exit node for that packet in that topology have to be recorded in the added fields. This has to be done before the packet is forwarded.

A packet with the extended forwarding header is either a transit or an outgoing packet at subsequent nodes in the forwarding topology that the packet follows. If, at a subsequent node, a comparison of the identifier of the exit node in the packet's forwarding header with the identifier of the node confirms that they are identical, the packet is an outgoing, i.e. a non-transit packet at that node. Otherwise, it is a transit packet. The forwarding instruction for the packet is then retrieved from the corresponding table with forwarding instructions. The additional fields which were added to the forwarding header of the packet at the entry node have to be removed before the forwarding instruction for the outgoing packet is retrieved.

The table with forwarding instructions for transit packets is a copy of the forwarding table of that node. An instruction contains the mappings:

forwarding topology $\rightarrow$ *outgoing edge of the node.*

It indicates the outgoing edge of the node where a packet which follows the specified forwarding topology, according to the identifier of the forwarding topology in the packet's forwarding header, has to be forwarded.

An alternative method of determining when a packet has reached the exit node involves tracking the number of hops to that exit node. This requires that for every entry node, the number of hops from that node to every exit node in every forwarding topology that contains it must be determined. The forwarding process is then as follows. The number of hops to the exit node, rather than the exit node identifier, is recorded in the forwarding header of a packet before that packet is forwarded at the entry node. The remaining hop count is decremented at each subsequent node and compared to zero. Once

zero is reached, the packet is an outgoing packet at that node. Otherwise it is a transit packet. The outcome indicates which of the two tables with forwarding instructions contains the forwarding instruction for that packet.

The rules for defining uniquely the identifiers of forwarding topologies depend on the routing strategy used and are open for modifications. In hop-by-hop IP routing, for example, where directed trees are used as forwarding topologies, and where each directed tree in use has a different root node, each forwarding topology can be distinguished by the identifier of its root node, i.e. $ForwardingTopologyID = RootNodeID$. In this case the root node of a forwarding topology is also the exit node for the packets which follow that topology, so a single field for recording the exit node identifer and the forwarding topology identifer is sufficient in the forwarding header. Directed trees are also used as forwarding topologies in multi-topology routing, but then more than one directed tree forwarding topology may exist which share the same root node. Additional information, beside the root node identifier, is then needed to distinguish each forwarding topology in use, e.g. $ForwardingTopologyID = [SequenceNumber, RoodNodeID]$. In constraint-based routing the forwarding topologies used are directed lines, and a number of them may share the same root node and the same leaf node. Following the above principles, the identifier of a forwarding topology in use could then be, e.g. $ForwardingTopologyID = [SequenceNumber, LeafNodeID, RoodNodeID]$. Again, the root node identifier and the exit node identifier are identical, and that can be used for reducing the information carried in the forwarding header. When forwarding topologies in use are managed centrally the identifiers with no internal structure can be used.

Note: As there are less then 256 nodes in an IP area network, a single byte is sufficient to identify the network nodes.

Overall, the forwarding protocol proposed in this section has the same advantages over IP as has MPLS, that is, it allows for:

- faster forwarding header examination, and
- faster lookup of tables with forwarding instructions,

and it also has a number of advantages over MPLS, in particular:

- hop-by-hop header modifications in the forwarding process of transit packets are avoided,
- the overhead of label distribution is avoided, and
- the need for modifying the label binding scheme depending on the routing strategy deployed is avoided.

5.2 The Control Protocol

The control protocol provides a centralised mechanism for controlling factors that affect traffic routes. Its functions include:

- the establishment of forwarding topologies of any type,
- the use of routing topologies in management of forwarding topologies, and
- the specification of routing assignments:
  - by activating appropriate network protocol algorithms,
  - by explicitly defining a particular routing assignment, and
  - by modifying sets of addresses specified as reachable from network nodes.

The control protocol uses flooding in distributing topology information to network nodes. The forwarding and routing topologies in use are defined compactly in respect to the network topology. This compact record is a flag for each network edge which indicates if that edge exists or not in the defined topology. The protocol distributes the network topology information initially. Subsequently, when a need for a particular forwarding or routing topology arises, its compact information is distributed network wide. This is a fast and reliable solution which introduces minimum information overhead when compared to the solutions used in IP and MPLS networks described in section 4.3.

5.3 Routing Strategies

The above described protocols facilitate the deployment of a range of new routing strategies. A simple example of such a routing strategy, called ordered routing, is given below.

In ordered routing each forwarding layer that has been established is defined by its type (point-to-point, point-to-multipoint, multipoint-to-point, multipoint-to-multipoint) and its forwarding topology. The forwarding topologies are either directed lines or directed rings.

The order of establishment of forwarding layers is defined, either for all routing states, or for particular network states. It is the order of assignment of destination addresses to the forwarding topologies in a defined ordered set of forwarding topologies, which is done at each entry node. At an entry node, the destination addresses reachable from an exit node are assigned to the first forwarding topology in the ordered set which provides connectivity to the exit node. If, for example, a link or a node failure means that some destinations are no longer reachable using that topology, they are assigned to the next forwarding topology in the ordered set with the necessary connectivity. Hence traffic is rapidly diverted to valid routes when, e.g., a link fails.

A number of illustrative examples follow.

Suppose that in Fig. 5.1a) every single network edge is defined as a single point-to-point forwarding layer, and that these single hop layers are always established first. The two directed rings in the clockwise and anticlockwise directions in the highlighted ring topology become the F_c and F_a multipoint-to-multipoint forwarding layers, respectively. Parameter μ of their multipoint-to-multipoint routing algorithm is defined as $\mu = (n-1)/2$. Provided that the assignment of destination addresses to forwarding topologies is done in the following order: $\{$single edge forwarding topologies, F_c, $F_a\}$, all one hop traffic follows the network topology, and all remaining traffic is split between the two directed rings.

Let the directed ring and line in the clockwise direction in the routing topology in Fig. 5.1b) be F_c^I , and the directed ring and line in the anticlockwise direction be F_a^I, all multipoint-to-multipoint forwarding layers, with $\mu = 2$. By using F_c^I and F_a^I in front of F_c and F_a in the ordered list, traffic between the nodes they contain that are two hops away

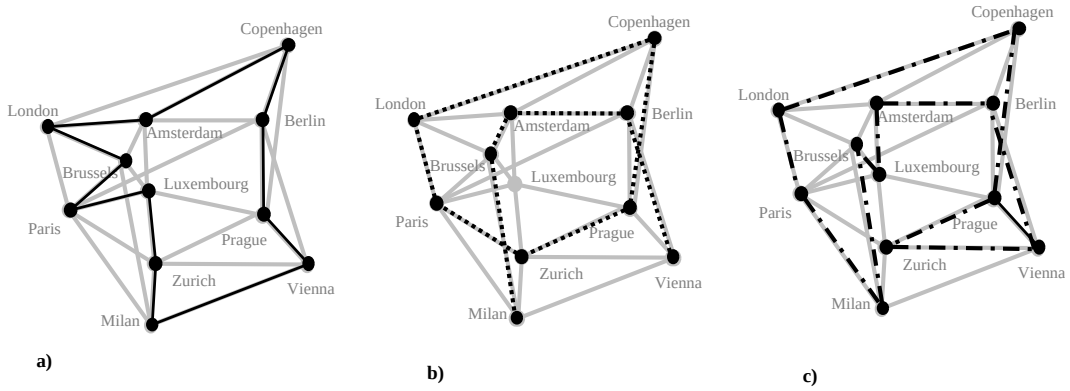


Figure 5.1: An Example of Ordered Routing

is diverted from F_c and F_a to the newly introduced layers.

If the two directed rings in Fig. 5.1c) are added as multipoint-to-multipoint forwarding layers F_c^{II} and F_a^{II} with $\mu = (n-1)/2$ at the end of the list, any failure of a single network element in F_c and F_a will result in the affected traffic on these rings being diverted onto F_c^{II} and F_a^{II}.

5.4 The Link Mask Topology and Link Cost Topology Algorithms

Two new routing topology algorithms are introduced below. The purpose of a routing topology algorithm is to determine a set of routing topologies for packet-switched networks, based on some traffic engineering criterion. The two algorithms maximize network throughput while maintaining fairness in distribution of network resources. They are called the Link Mask Topology (LMT) algorithm, and the Link Cost Topology (LCT) algorithm.

The two algorithms differ in following different sets of rules for the allocation of resources to multiple paths between a pair of network nodes which may arise from the routing topologies generated by the algorithm. Their common features are described below.

The input data for both algorithms is the network topology and the demand matrix. The network topology comprises the set of network nodes and edges, and the capacities of the network edges. The demand matrix contains the values of the expected traffic

demands between every pair of network nodes. This might, for example, be generated offline from historical records of network traffic.

The output of the two algorithms is a set of routing topologies and a capacity matrix for each routing topology in that set. The capacity matrix of a routing topology specifies the capacities of all the paths available in that routing topology. The paths applicable in routing are those with non-zero capacity only.

The set of routing topologies generated always includes one with a set of network elements matching those of the input network topology, where the length of each edge is inversely proportional to the capacity of the corresponding network edge. That routing topology is referred to as the base topology. It is identical to the single routing topology that is most commonly used in hop-by-hop IP routing.

The paths that an output routing topology provides are the shortest paths in that topology, as determined by using Dijkstra's algorithm. Any ties in calculation of path costs during the calculation of shortest path tree are resolved based on the node identifiers, the node with the lower identifier being preferred.

Before discussing the details of each algorithm, their common approach to bandwidth allocation will be described.

5.4.1 Bandwidth Allocation in the LMT and LCT Algorithms

The LMT and LCT algorithms allocate resources in a topology to a set of paths by using a novel resource allocation approach called prioritized max-min bandwidth allocation. The idea for this research allocation approach comes from the max-min bandwidth allocation scheme initially proposed in [106]. Further research on that topic can be found in [107, 108].

Max-min Bandwidth Allocation

Max-min bandwidth allocation is a resource allocation technique which allocates resources to paths in a topology in an equal share while it fully utilises the edge capacities whenever possible. The term *max-min* indicates that this technique *maximizes* the bandwidth allocated to the paths that receive the *minimum* bandwidth among all paths. The

following are two of its basic properties.

- At each edge e, every path on that edge, $p \in P(e)$, is allocated an equal share of the edge capacity. A path p may thus be allocated different bandwidths at the edges it passes. The edge where the minimum bandwidth is allocated to that path is its bottleneck edge. It dictates the resulting bandwidth allocated to path p.

- The entire capacity of edge e is allocated to the paths that pass that edge, $P(e)$, unless one or more paths, $p \in P(e)$, have a bottleneck edge elsewhere which limits the bandwidth that path p can receive to a lower value.

The resource allocation is performed as follows. The maximal minimal (*max-min*) bandwidth of each path is determined by the path's bottleneck edge. A global bottleneck edge e_b is the edge with the smallest bandwidth per path. As such it is the bottleneck for each path that passes that edge, $p \in P(e_b)$. Initially, an equal share of edge e_b capacity, c_{e_b}, is allocated to all the paths that pass that edge, that is $\frac{c_{e_b}}{|P(e_b)|}$. Subsequently, all of these paths are removed from the topology by reducing the capacities of the edges they pass by the value allocated to the paths, $\frac{c_{e_b}}{|P(e_b)|}$. The above procedure is then repeated until resources are allocated to every path and all paths are removed from the topology.

Prioritized Max-min Bandwidth Allocation

Prioritized max-min bandwidth allocation is a resource allocation technique which allocates resources to paths in a topology in proportion to the traffic demands on each path while seeking to fully utilise the edge capacities. The novelty introduced here when compared to the standard max-min bandwidth allocation is in the following.

- At each edge e, every path on that edge $p_{i,j}$ between nodes i and j is allocated a share of the edge capacity, in proportion to the traffic demands on that path, $d_{i,j}$. The edge where the minimum bandwidth is allocated to a path determines the resulting bandwidth allocated to that path.

The following property is shared with the earlier resource allocation technique.

- The entire capacity of edge e is allocated to the paths that pass that edge, $P(e)$, unless one or more paths, $p \in P(e)$, have a bottleneck edge somewhere else which limits the bandwidth that path p can receive to a lower value.

The prioritized max-min resource allocation is done as follows. The maximal minimal (*max-min*) bandwidth of each path is determined by the path's bottleneck edge. The global max-min bandwidth, b_{mm}, is determined by the global bottleneck edge e_b , which is the edge that allows the smallest max-min bandwidth for the set of paths $P(e_b)$ on that edge. Assuming that the capacity of edge e_b is c_{e_b}, b_{mm} is determined by the following expression:

$$b_{mm} = \frac{c_{e_b}}{\sum_{\forall p_{i,j} \in P(e_b)} d_{i,j}}$$

where $d_{i,j}$ are the demands on paths $p_{i,j}$. Capacities and demands may be expressed in bandwidth units. Initially, the entire capacity of the global bottleneck edge is allocated to the paths traversing that edge. The bandwidth allocated to path $p_{i,j}$ is:

$$b_{i,j} = d_{i,j} b_{mm}$$

All the paths with allocated bandwidth are subsequently removed from the topology by reducing the capacities of the edges these paths traverse by the value allocated to the paths. The above process is then repeated until resources are allocated to all paths, and all paths are removed from the topology.

5.4.2 The LMT Algorithm

Traditionally in hop-by-hop IP routing traffic is routed along shortest paths. While this approach conserves network resources, it may also critically limit network throughput. For example, a number of shortest paths may overlap on a network edge, causing congestion on that edge, or the capacity of the shortest path may be insufficient for the demands on that path. At the same time alternative paths may exist with sufficient capacity to take over some traffic on the critical shortest paths. The aim of the Link Mask Topology

(LMT) algorithm is to find such paths and to assign some traffic on the shortest paths to the found paths, so that congestion is avoided and network throughput is increased.

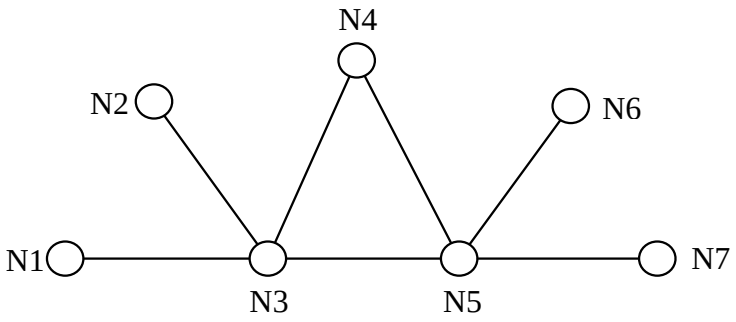
Thus the LMT algorithm finds additional *longer* paths in a set of routing topologies it defines. These routing topologies are defined based on the input network topology and the estimated traffic demands so that they do not contain critical network edges. Critical edges are the edges where congestion may be expected when traffic is routed exclusively along shortest paths through the network. The additional paths that the LMT algorithm provides are the shortest paths in a set of routing topologies that the algorithm defines. However, as these topologies contain only a subset of network edges, the paths they provide are not necessarily the shortest paths through the network.

A simplified example of the operation of the LMT algorithm is presented in Fig. 5.2. In the network shown all network edges have unit capacities, and unit lengths. Traffic demands $d_{1,7}$, between nodes $N1$ and $N7$, and demands $d_{2,6}$, between nodes $N2$ and $N6$, are non-zero and equal. The demands between all other pairs of nodes are zero.

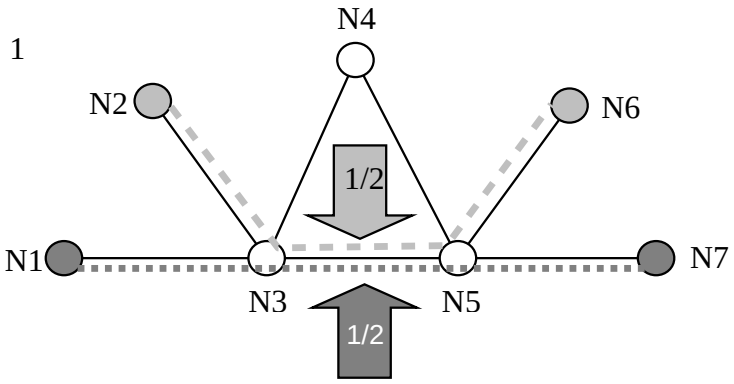
In step 1 in Fig. 5.2 the network topology is used as a routing topology. All traffic is routed along the shortest paths through the network, and the resulting traffic distribution is shown. The two shortest paths used overlap on edge $e(3,5)$, which in turn limits the capacities of the two paths to 0.5 units of bandwidth. This edge is the global bottleneck edge in the network, as no other edge limits the capacities of the two paths to a lower value.

In step 2 an additional routing topology, topology 2, is created which does not contain the identified global bottleneck edge. The shortest paths for the pairs of nodes with non-zero traffic demands are found in this topology. Traffic on path $p_1(1,7)$ between nodes $N1$ and $N7$ in the network, i.e. in the first routing topology, is then assigned to path $p_2(1,7)$ in topology 2. The new traffic distribution across two routing topologies is shown in the figure. It can be seen that this new distribution allows for one bandwidth unit to be allocated for the two pairs of nodes with non-zero traffic demands. That is twice the capacity than these pairs of nodes could have been allocated when only the shortest network paths were used.

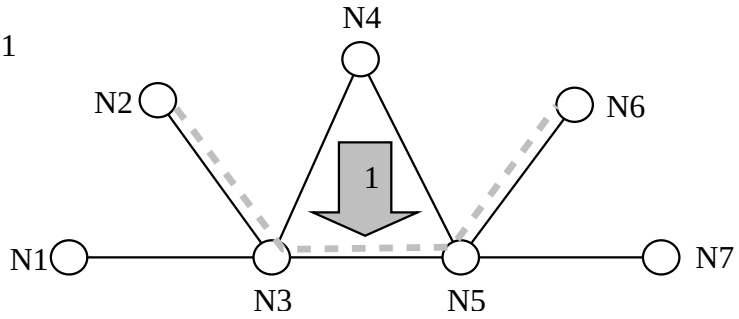
Network:



Step 1:
Edge lengths: 1



Step 2:
Topology 1:
Edge lengths: 1



Topology 2:
Edge lengths: 1

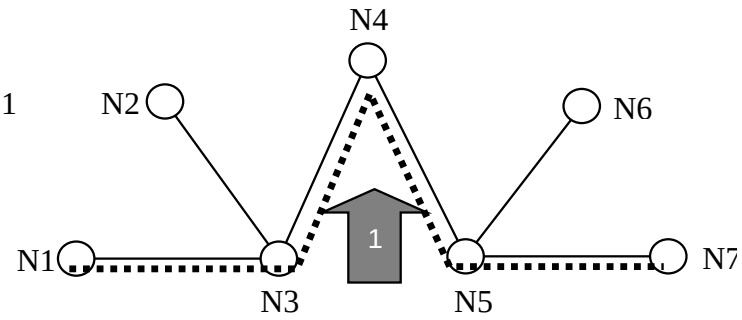


Figure 5.2: The LMT Algorithm

If the new traffic distribution had shown that no improvement in resource allocation could have been achieved by shifting traffic from path $p_1(1,7)$ to path $p_2(1,7)$, then an attempt would have been made to shift traffic from path $p_1(2,6)$ to path $p_2(2,6)$. No traffic shifts are made unless an increase in the resources allocated to the pairs of nodes with non-zero demands is confirmed.

A detailed description of the operation of the LMT algorithm operation is given below.

The LMT algorithm is initialized by creating the base topology, $T_{k,k=1}$, based on the input network topology, T. The base topology has a set of nodes and edges identical to the network topology. The length of each edge is inversely proportional to the edge capacity. The shortest paths between the nodes with non-zero traffic demands in the base topology are initially marked as active paths.

The algorithm starts by allocating network resources to active paths and then locating global bottleneck network edges. The bandwidth $b_{i,j}$ of the active path $p_{i,j}$ between nodes i and j, is:

$$b_{i,j} = d_{i,j} b_{mm}$$

where b_{mm} is the maximal minimal (*max-min*) bandwidth, and $d_{i,j}$ are the bandwidth demands between nodes i and j. Max-min bandwidth b_{mm} is determined by the global bottleneck edge e_b , i.e. by the edge that allows the smallest b_{mm} for the set of path $P(e_b)$ on that edge. Assuming that the capacity of edge e_b is $c(e_b)$, b_{mm} is determined by the following expression:

$$b_{mm} = \frac{c_{e_b}}{\sum\limits_{\forall i,j \in P(e_b)} d_{i,j}}$$

The utilisation u_e of every edge e is subsequently determined as the ratio of the bandwidth that has been allocated to the active paths and the total capacity of that edge. That is, utilisation u_e is calculated according to the following expression:

$$u_e = \frac{\sum\limits_{\forall i,j \in P(e)} d_{i,j} b_{mm}}{c_e}$$

where $P(e)$ defines the set of paths $p_{i,j}$ on edge e, and c_e is the capacity of edge e. Edge e is considered to be *critical* if the proportion of network edges which have a lower utilisation exceeds η, where η is a configurable parameter in the range $0 < \eta \leq 1$.

After the critical network edges have been determined, a new routing topology, T_{k+1}, is created by removing these edges from the base topology. The algorithm aims to increase the max-min bandwidth b_{mm} by deactivating the paths in the previously created routing topologies (which contain the critical edges), and by activating the corresponding paths, if they exist, in the new routing topology (which does not contain the critical edges). There is a set of *the LMT algorithm* conditions that must be fulfilled so that a currently active path can be replaced by an alternative path. They are listed below.

The LMT Algorithm Conditions:

A *new* path, $p_{i,j}(T_{k+1})$, between nodes i and j in topology T_{k+1}, if available, is activated, and an *old* path, $p_{i,j}(T_{l,l\in[1,k]})$, between the two nodes in a previously created routing topology is deactivated, if and only if:

- the *old* path, $p_{i,j}(T_{l,l\in[1,k]})$, contains a critical edge,
- when traffic on the *old* path, $p_{i,j}(T_{l,l\in[1,k]})$, is redirected to the *new* path, $p_{i,j}(T_{k+1})$, no edge on that *old* path has a higher utilisation than the most utilised edge on the *new* path, and
- the *new* path, $p_{i,j}(T_{k+1})$, is no more than Δh_{max} hops longer than the *old* path, $p_{i,j}(T_{l,l\in[1,k]})$,

where Δh_{max}, a positive integer, is a configurable parameter.

Applying these conditions ensures that a new path is activated, and the old one is deactivated, only if such changes do not lead to a worse traffic distribution in the network. The third condition above is introduced to prevent the selection of excessively long paths, as such paths consume excessive resources and increase delay.

The paths are processed sequentially. The paths which originate from a node with a lower identifier are processed prior to those which originate from a node with a higher

identifier. The paths which terminate at a node with a lower identifier are processed prior to those which terminate at a node with a higher identifier.

The above procedure is repeated until no further increase in max-min bandwidth b_{mm} can be achieved, i.e. $\Delta b_{mm} = 0$, or the maximum number of routing topologies, k_{max}, has been reached. Network bandwidth is then allocated to active paths by using the prioritized max-min bandwidth allocation described in section 5.4.1.

The operation of the LMT algorithm is summarized below.

The LMT Algorithm	
INPUT:	Network Topology: T Traffic Demands: D
OUTPUT:	Routing Topologies: T_1, T_2, ... , T_k Capacities of Paths: C_{T_1}, C_{T_2}, ..., C_{T_k}
PROCEDURE:	k = 1; $T_k = T$; Mark as *active* path $p_{i,j}$ in topology T_k if demand $d_{i,j} > 0$; 1. Determine critical edges; 2. Create T_{k+1} by removing critical edges from T_k; 3. *Deactivate* paths in $T_{l,l\in[1,k]}$ which contain critical edges, *activate* corresponding paths with no critical edges in T_{k+1} if and only if the LMT algorithm conditions are fulfilled; 4. k++; Repeat steps 1 to 3 while $(k < k_{max})$ and $(\Delta b_{mm} > 0)$; 5. Allocate bandwidth to active paths by using prioritized max-min bandwidth allocation.

A note on the name of the algorithm. In a packet-switched network the nodes and edges are routers and links, respectively. The LMT algorithm creates a set of routing topologies by removing a subset of network edges from the base network topology. That is, this algorithm *masks* a number of *links* in the network and that way it creates a set of routing *topologies*. Hence it is called the *Link Mask Topology* (LMT) algorithm.

5.4.3 The LCT Algorithm

As the growth in demand has made network service less reliable, network users have expressed a readiness to pay more for a guaranteed quality of service. In a request for such a guarantee, a user might be asked to specify the amount of network resources that it needs. An agreement could then be reached between the network provider and the user,

which obliges the network provider to reserve the specified amount of network capacity for that user, but which also obliges the user to ensure that the traffic it generates is within the requested boundaries. Knowledge of the actual sizes of individual traffic flows can be used to infer the distribution of traffic across set multiple paths. The Link Cost Topology (LCT) algorithm relies on the existence of such information. Like the LMT algorithm, the LCT algorithm looks for alternative paths which could potentially accept some traffic from the shortest paths, so that congestion is avoided and network throughput is increased. However, the LCT algorithm assigns resources to a number of paths between any two network nodes with non-zero traffic demands. In doing so this algorithm directs traffic between any such pair of nodes to multiple paths, whereas the LMT algorithm directs this traffic only to a single path.

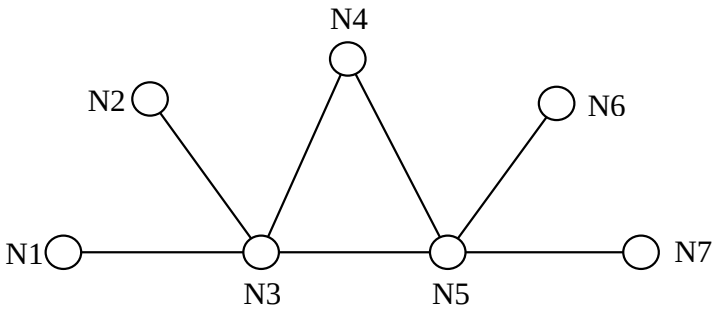
The LCT algorithm finds additional *longer* paths in a set of routing topologies it defines. These routing topologies are defined based on the input network topology and the estimated traffic demands so that the length of any critical network edge is increased.

A simplified example of the operation of the LCT algorithm is given in Fig. 5.3. All network edges in the example network have unit capacities, and unit lengths. Traffic demands $d_{1,7}$, between nodes $N1$ and $N7$, and demands $d_{2,6}$, between nodes $N2$ and $N6$, are non-zero and equal. The demands between any other two nodes are zero.

In step 1 the network topology is used as a routing topology. The resulting traffic distribution when all traffic is routed along the shortest paths through the network is shown in Fig. 5.3. As the two used shortest paths overlap on edge $e(3,5)$, the capacity of these two paths is limited to 0.5 units of bandwidth by the unit capacity of edge $e(3,5)$. This edge is the global bottleneck edge in the network, as no other edge limits the capacities of the two paths to a lower value. The *maximal minimal* capacity of 0.5 bandwidth units is allocated for the two paths in the routing topology in step 1. The residual capacity is then calculated for each edge as the difference in the total edge capacity and the value of the edge capacity allocated to the two paths.

In step 2 an additional routing topology, topology 2, is created. This topology contains a set of nodes and edges identical to the initial routing topology, i.e. the network topology,

Network:



Step 1:

Edge lengths: 1

Residual capacities:

$c_r(1,3) = 1/2$

$c_r(2,3) = 1/2$

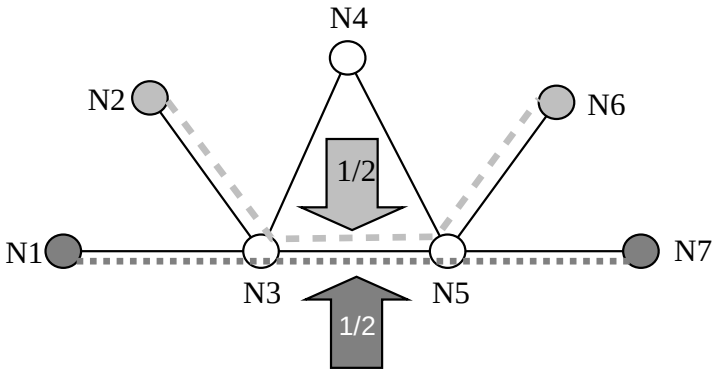
$c_r(3,4) = 1$

$c_r(3,5) = 0$

$c_r(4,5) = 1$

$c_r(5,6) = 1/2$

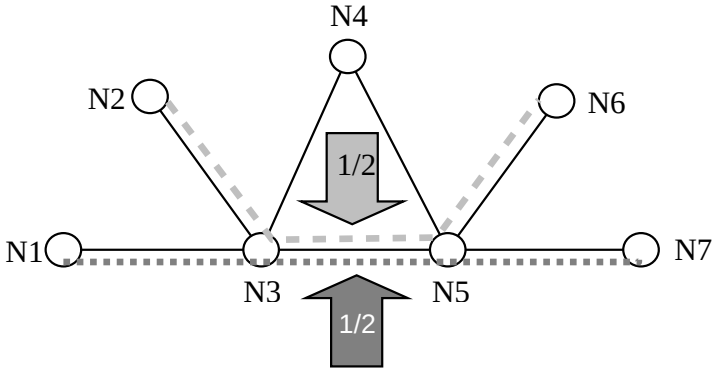
$c_r(5,7) = 1/2$



Step 2:

Topology 1:

Edge lengths: 1



Topology 2:

Edge lengths:

l(1,3) = 2

l(2,3) = 2

l(3,4) = 1

l(3,5) = inf

l(4,5) = 1

l(5,6) = 2

l(5,7) = 2

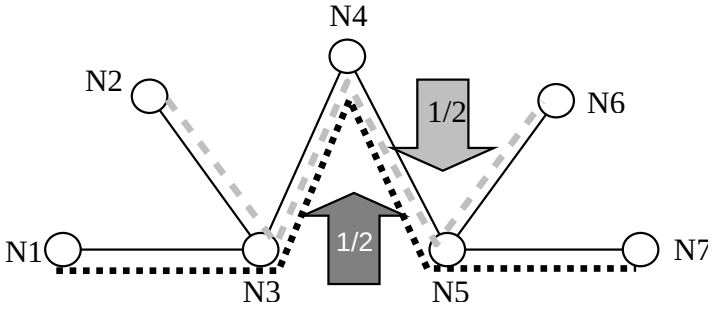


Figure 5.3: The LCT Algorithm

but the edge capacities and their lengths are different. The capacity of each edge is set equal to the residual capacity of that edge in the initial routing topology, as calculated in step 1. The edge length is inversely proportional to the edge capacity. The resulting edge lengths in topology 2 are listed in the figure, and the shortest paths in topology 2 for the two pairs of nodes which have non-zero traffic demands are shown. The two shortest paths now overlap on edges $e(3,4)$ and $e(4,5)$. The unit capacities of the two edges in topology 2 limit the capacity of each path to 0.5 bandwidth units. All the other edges which these two paths pass limit the capacities of the two paths to this same value, so there are six global bottleneck edges in this case. The newly found *maximal minimal* capacity of 0.5 bandwidth units is allocated for the two paths in topology 2.

With 0.5 units of bandwidth allocated to the two pairs of nodes with non-zero traffic demands in the two defined routing topologies, there is in total one unit of bandwidth for each such pair of nodes. That is twice the capacity that could have been assigned to the two pairs of nodes when only the shortest network paths were used. Traffic between a pair of nodes can now be split across two paths. When only the shortest paths are used that traffic follows a single path only.

A detailed description of the LCT algorithm operation follows.

The LCT algorithm is initialized by creating the base topology, $T_{k,k=1}$, based on the input network topology, T. This topology has a set of nodes and edges identical to the network topology. The length of each edge is inversely proportional to the edge capacity.

The algorithm starts by allocating a bandwidth:

$$b_{i,j} = d_{i,j} b_{mm}(k)$$

to the shortest paths in T_k. Bandwidth $b_{mm}(k)$ is the max-min bandwidth dictated by the global bottleneck edge $e_b(k)$ in T_k, which is the edge that allows the smallest value of $b_{mm}(k)$ for the set of path $P(e_b(k))$ on that edge in T_k. Assuming that the capacity of edge

$e_b(k)$ in T_k is $c_{e_b}(k)$, $b_{mm}(k)$ is determined by the following expression:

$$b_{mm}(k) = \frac{c_{e_b(k)}}{\sum_{\forall i,j \in P(e_b(k))} d_{i,j}}$$

The residual capacities of edges in topology T_k are then determined, and a new routing topology, T_{k+1}, is created. Topology T_{k+1} has a set of nodes and edges identical to topology T_k, but the edges in the two topologies have different capacities and lengths. Each edge in topology T_{k+1} has a capacity equal to the residual capacity of the corresponding edge in topology T_k. It is determined by the following expression:

$$c_e(k+1) = c_e(k) - \sum_{\forall i,j \in P(e)} d_{i,j} b_{mm}(k)$$

where $c_e(k+1)$ is the capacity of edge e in topology T_{k+1}, and also the residual capacity of that edge in T_k, $c_k(e)$ is the capacity of edge e in topology T_k, and $P(e)$ is the set of paths $p_{i,j}$ on edge e in T_k. The length of edge e in topology T_{k+1} is equal to the reciprocal value of the edge capacity in T_{k+1}, that is:

$$l_e(k+1) = \frac{1}{c_e(k+1)}$$

The above procedure is repeated for the newly created topology, T_{k+1}. So the max-min bandwidth $b_{mm}(k+1)$ of the global bottleneck edge $e_b(k+1)$ is determined in topology T_{k+1}, and a bandwidth $b_{i,j} = d_{i,j} b_{mm}(k+1)$ is allocated to the shortest paths in that topology. Topology T_{k+2} is then created, and another cycle of the same procedure starts.

The described procedure is repeated for as long as the max-min bandwidth $b_{mm}(k+i)$ determined for topology T_{k+i} exists, i.e. while $b_{mm}(k+i) > 0$, or until the maximum number of routing topologies, k_{max}, has been generated. In the final topology, T_{k+i}, capacities are assigned to paths by using the prioritized max-min bandwidth allocation process described in section 5.4.1. That is, the resources in topology T_{k+i} are allocated to the paths in T_{k+i} by the LCT algorithm in proportion to traffic demands on each path.

The operation of the LCT algorithm is summarized below.

The LCT Algorithm	
INPUT:	Network Topology: T Traffic Demands: D
OUTPUT:	Routing Topologies: T_1, T_2, ... , T_k Capacities of Paths: C_{T_1}, C_{T_2}, ..., C_{T_k}
PROCEDURE:	k = 1; $T_k = T$; 1. Determine max-min bandwidth $b_{mm}(k)$ in topology T_k, and allocate bandwidth $d_{i,j}b_{mm}(k)$ to shortest paths $p_{i,j}$ in T_k; 2. Create topology T_{k+1} where: edge e capacity: $c_{k+1}(e) =$ residual $c_k(e)$ in T_k, edge e length: $l_{k+1}(e) = 1/c_{k+1}(e)$. 3. k++; Repeat steps 1 to 2 while $(k < k_{max})$ and $(b_{mm}(k) > 0)$; 4. Allocate bandwidth to paths provided in T_k by using prioritized max-min bandwidth allocation.

A note on the name of the algorithm. The network nodes and edges in packet-switched networks are routers and links, respectively. The LCT algorithm creates a set of routing topologies by modifying the lengths of edges in the base network topology. The edge length is commonly referred to as the *link cost*. Therefore, this algorithm changes the *costs* of a number of *links* in the network and so it creates a set of routing *topologies*. Hence it is called the *Link Cost Topology* (LCT) algorithm.

5.5 Summary

A novel forwarding protocol has been described in this chapter, which is inspired by MPLS but avoids the overhead of label binding and distribution. A simple control protocol has also been described which allows new routing strategies to be deployed. Two routing topology algorithms for traffic control purposes have been presented: the LMT algorithm and the LCT algorithm. These algorithms aim to prevent congestion and to increase the throughput in networks beyond that achievable with shortest path routing by discovering alternative paths for some traffic which would alternatively be routed along the shortest paths. The LMT algorithm ensures that traffic between any two network nodes follows a single path, whereas the LCT algorithm splits traffic between any two network nodes onto a number of paths. So, the LCT algorithm relies on the existence of more sophisticated traffic tuning techniques in the network than the LMT algorithm. Two simple examples of these algorithms were given in this chapter to show that they can

significantly improve network throughput while maintaining fairness in resource allocation. However, an accurate evaluation of the performance of the two algorithms requires extensive tests on different networks. The results of a set of executed performance tests of the LMT and LCT algorithms will be presented and discussed in the next chapter.

Chapter 6

Evaluation of the LMT and LCT algorithms

Two routing topology algorithms, called the LMT algorithm and the LCT algorithm have been introduced in chapter 5. The two algorithms define a set of routing topologies, and select routes in the defined topologies. A key performance objective of these algorithms is to maximize network throughput while maintaining fairness in distribution of network resources. The performance of the two proposed algorithms has been compared to the shortest path routing algorithm by their author through calculations in a set of test cases. The results obtained in a number of test cases are presented and discussed in this chapter.

6.1 The Network Model

Two reference networks have been used in the evaluation of the LMT and LCT algorithms. They are shown in Fig. 6.1 and Fig. 6.2. The performance of the routing algorithm proposed in [87, 88, 89, 90] has been evaluated on the network shown in Fig. 6.1. The performance of the LMT and LCT routing topology algorithms on this network has also been investigated. The example in Fig. 6.2 has been used in the performance evaluation of routing algorithms in [109, 110, 111], where this network topology is referred to as the Internet Service Provider (ISP) topology. The results of the LMT algorithm for a set of tests performed on this network have been published in [53]. They are shown in section 6.3.4.

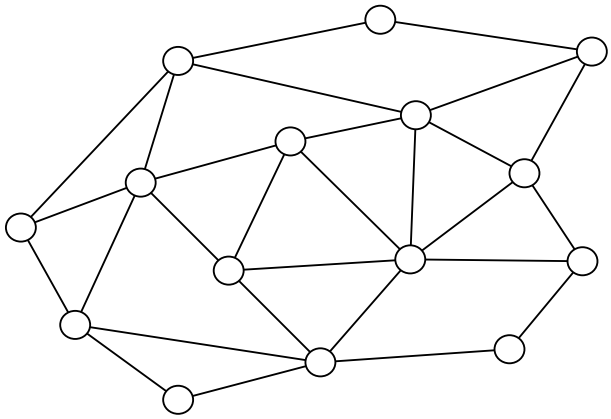


Figure 6.1: Network Example 1

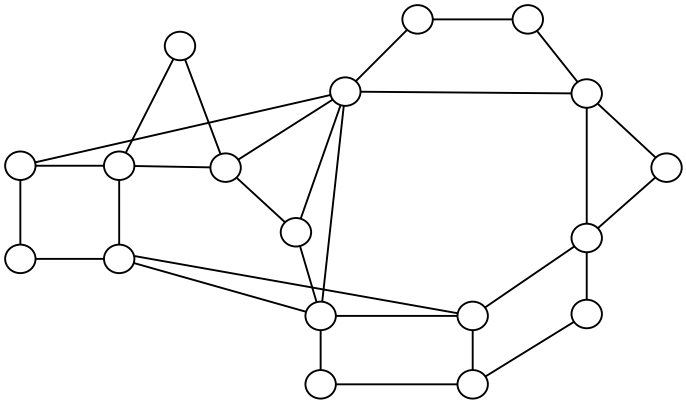


Figure 6.2: Network Example 2 (*The ISP Topology*)

6.2 Performance Metrics and Representation

Performance evaluation of the LMT algorithm and the LCT algorithm has been done by monitoring the bandwidth allocated by the two algorithms to each pair of nodes which has non-zero traffic demands in the network under study. Such a pair of nodes will be referred to as an active node pair in the following. The allocated bandwidth values change depending on the number of routing topologies that each algorithm defines.

The algorithms aim to simultaneously achieve:

- high throughput,
- fairness in assigning the network bandwidth, and
- high utilisation of network bandwidth.

These values are measured by observing the overall utilisation of network bandwidth,

and a set of metrics which describes the bandwidth values allocated to each active node pair. This set of metrics includes:

- the minimum allocated value,
- the median value, which is the smallest value in the ordered set of allocated values such that at least half of the remaining values are not greater than it,
- the lower quartile value, which is the smallest value in the ordered set of allocated values such that at most a quarter of the remaining values are lower than it,
- the upper quartile value, which is the largest value in the ordered set of allocated values such that at most a quarter of the remaining values are higher than it,
- the maximum allocated value,
- the median range, which is the difference between the upper quartile value and the lower quartile values and so contains at least half of the allocated values,
- the mean allocated value, i.e the average bandwidth allocated to active node pairs,
- the variance of allocated values, and
- the aggregate bandwidth allocated to all active node pairs.

The performance of either the LMT and LCT algorithm when using only one topology (the base topology) is equivalent to the performance of the shortest path routing algorithm. Hence comparing the capacities allocated to active node pairs in a number of output routing topologies and in the base topology only shows any improvement in performance over networks which operate with shortest path routing.

The metrics obtained will be summarized in a set of tables. The following values will also be presented graphically:

- the minimum allocated bandwidth value, and
- all the bandwidth values allocated to each active node pair.

Box plots are used to present the set of bandwidth values allocated to each active node pair. An example of such a box plot is shown in Fig. 6.3. The plot shows the minimum value (*min*), the lower quartile value ($q1$), the median value (*median*), the upper quartile value ($q2$), and the maximum value (*max*) in the monitored data set. The lower quartile value, $q1$, and the upper quartile, $q2$, define a box. This box contains at least half of the values in the monitored data set. The box size defines the median range since at most a quarter of the set membership lies in the ranges $(min, q1)$ and $(q2, max)$ respectively.

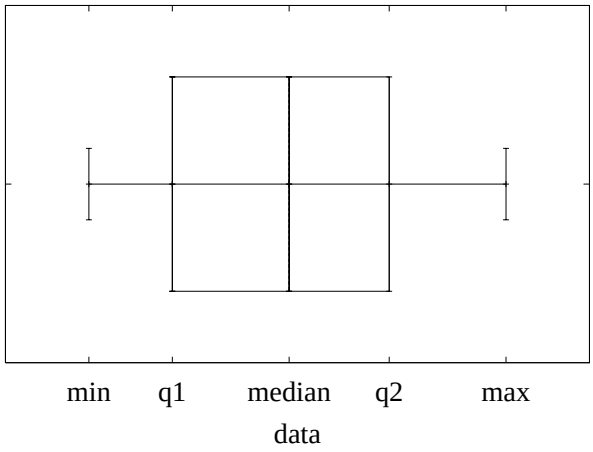


Figure 6.3: Box Plot

The box plot presentation is a convenient visual indication of the fairness of the routing topology algorithms under study. Larger boxes indicate less fair distributions.

6.3 Scenario 1: Balanced Load and Network

The first set of test results that are presented in this chapter is obtained by providing the following input information to the routing topology algorithms.

Network Topology: The input network topology is shown in Fig. 6.4. The shown topology consists of $n = 15$ nodes and $m = 28$ bidirectional edges. Each edge has a capacity of $C = 100$ bandwidth units.

Traffic Demands: Every two network nodes exchange an equal amount of traffic. The estimated traffic demand $d_{i,j}$ for any pair of nodes N_i and $N_{j,i\neq j}$ is $d_{i,j} = k$, where k is a positive value.

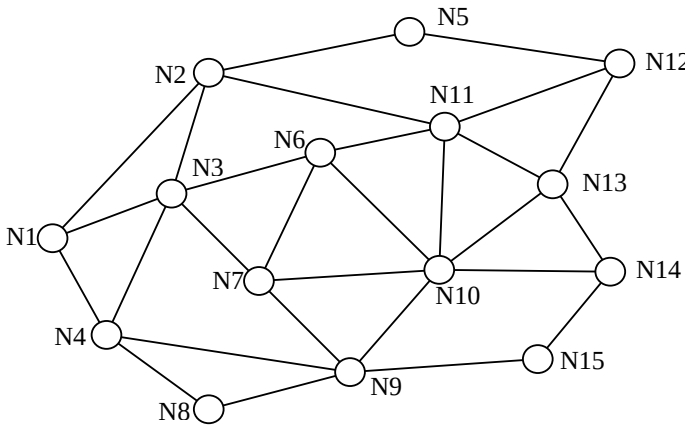


Figure 6.4: Network Topology, Scenario 1

Note that the enumeration of the nodes in Fig. 6.4 may affect the output of the algorithms since the node identifiers are used to resolve ties when multiple shortest paths exist (section 5.4).

The parameters η and Δh_{max} are set to 0.75 and 3 respectively. These parameters were defined in section 5.4.2.

6.3.1 The LMT Algorithm

The results obtained with the LMT algorithm in scenario 1 are presented graphically in Fig. 6.5 and Fig. 6.6. The numerical values of the observed metrics are given in Tables 6.1 and 6.2. The prioritized max-min bandwidth allocation technique used by this algorithm seeks to fully utilise network capacities whenever possible, and the network is fully utilised in this scenario.

Fig. 6.5 shows box plots of the bandwidth values allocated to the active node pairs depending on the number of routing topologies defined by the LMT algorithm. For clearer box plot presentation, extreme values of allocated bandwidth, above 20 units, are excluded. The figure shows an increase in fairness in bandwidth allocation, as the number of output routing topologies increases up to three. There is a reduction of the median range by 66% with two output routing topologies, and by 79% with three output routing topologies when compared to the case with a single output routing topology.

The minimum capacities allocated to the active node pairs depending on the number of routing topologies defined by the LMT algorithm are shown in Fig. 6.6. Starting from

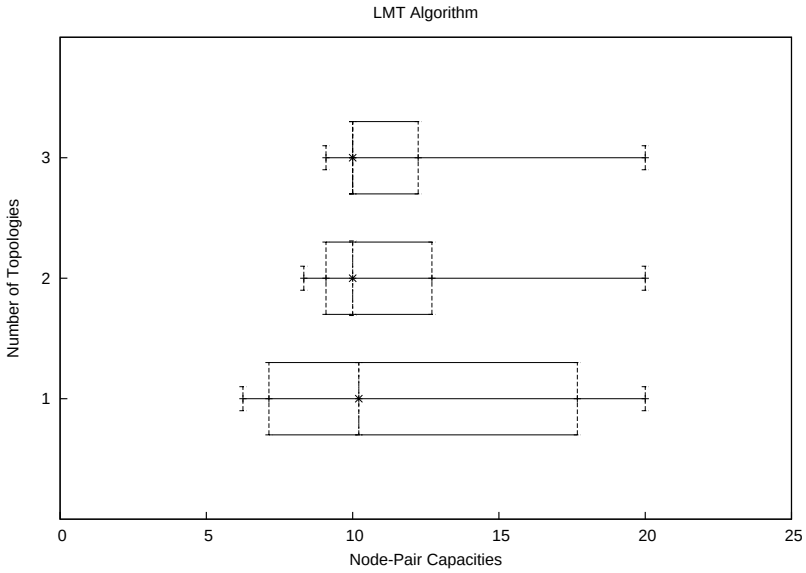


Figure 6.5: LMT Algorithm, Scenario 1 - Allocated Capacities

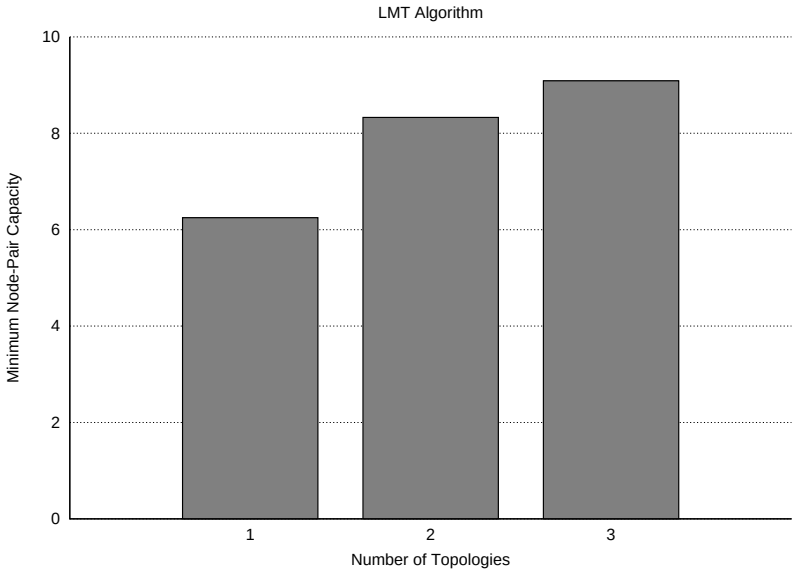


Figure 6.6: LMT Algorithm, Scenario 1 - Minimum Capacity

a single output routing topology, this minimum capacity is increased by 33% with one additional routing topology, and by 45% with two additional routing topologies defined by the LMT algorithm. No further increase of the minimum capacity was obtained by adding output routing topologies. When the number of routing topologies is three, the minimum capacity allocated to the active node pairs, and the minimum bandwidth allocated to a path, is 9.09 bandwidth units, much less than the edge capacity of 100 units. However, each edge will belong to multiple paths, so that the aggregate bandwidth of traffic on the edge approaches 100 units, or maximum utilisation.

	capacities for active node pairs (in bandwidth units)			
no. of topologies	minimum	median	maximum	median range
1	6.25	10.21	79.72	10.54
2	8.33	10	72.73	3.62
3	9.09	10	69.62	2.24

Table 6.1: LMT Algorithm, Scenario 1 - Box Plot Data

	capacities for active node pairs (in bandwidth units)		
no. of topologies	mean	variance	aggregate
1	14.88	11.93	3125.04
2	13.55	9.9	2844.61
3	13.22	8.85	2777.11

Table 6.2: LMT Algorithm, Scenario 1 - Additional Data

The LMT algorithm tends to fully utilise available capacities. As it creates more routing topologies, the algorithms shifts traffic from the shortest path to the longer ones. This leads to a decrease in the aggregate bandwidth allocated for active node pairs, as shown in Table 6.2.

6.3.2 The LCT Algorithm

The results obtained with the LCT algorithm in scenario 1 are shown in Fig. 6.7 and Fig. 6.8. The numerical values of the observed metrics are given in Tables 6.3 and 6.4. As the prioritized max-min bandwidth allocation technique used by this algorithm tends to fully utilise network capacities whenever possible, the network is again fully utilised in this scenario.

Fig. 6.7 shows box plots of the bandwidth values allocated to the active node pairs, depending on the number of routing topologies defined by the LCT algorithm. Once again, any allocated bandwidth values above 20 units are excluded. The figure shows a reduction of the median range by 74%, by 82%, and by 83%, with two, three, and four output routing topologies, respectively, when compared to the case with a single output routing topology. So the variations in bandwidth values allocated to at least half of the node pairs are reduced by 74%, by 82%, and by 83% in the three cases, respectively. Some of the allocated values are extremely high, so the variance of the observed data set in Table 6.4 does not capture these changes.

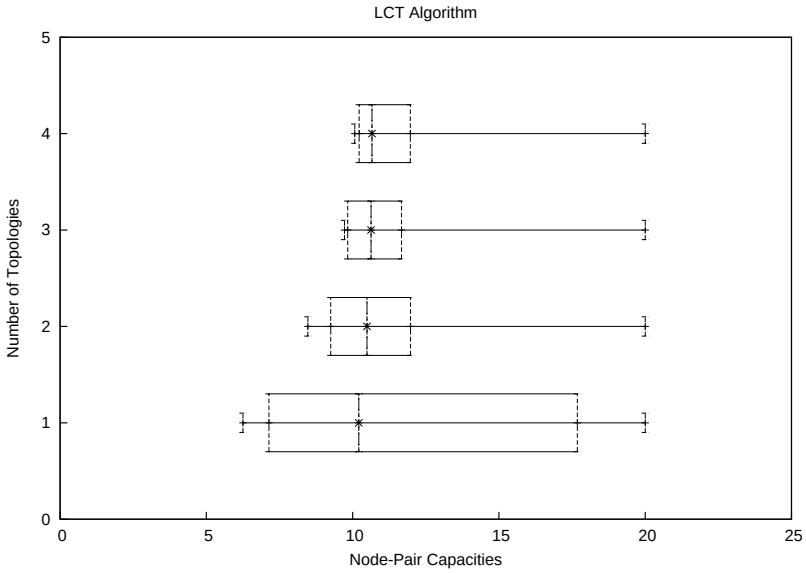


Figure 6.7: LCT Algorithm, Scenario 1 - Allocated Capacities

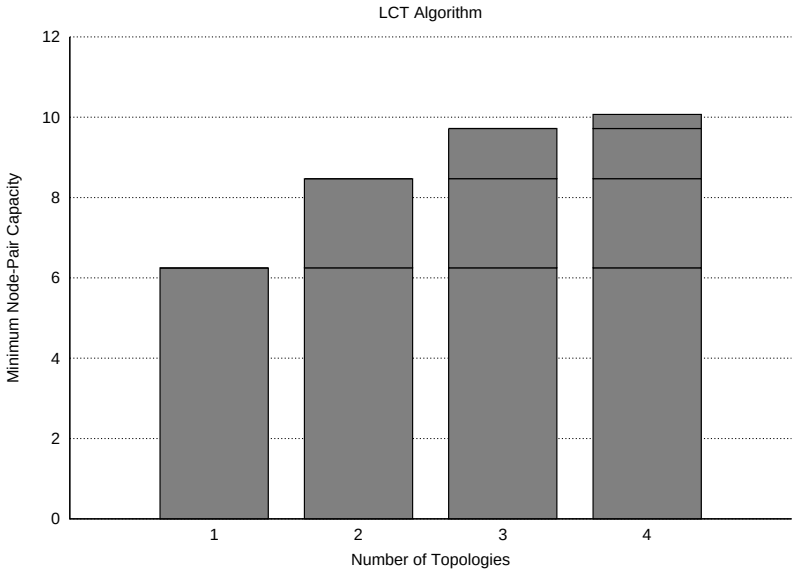


Figure 6.8: LCT Algorithm, Scenario 1 - Minimum Capacity

The minimum capacities allocated to the active node pairs depending on the number of routing topologies defined by the LCT algorithm are shown in Fig. 6.8. Starting from a single output routing topology, this minimum capacity is increased by 35% with one additional routing topology, by 56% with two additional routing topologies, and by 61% with three additional routing topologies defined by the LCT algorithm. The minimum capacities allocated in each of these topologies are given in Tab. 6.3 as *tpl. minimum*, and shown as increments in Fig. 6.8. They decrease by 64%, by 80%, and by 94% as the number of topologies increases from two to four.

	capacities for active node pairs (in bandwidth units)				
no. of topologies	tpl. minimum	minimum	median	maximum	median range
1	6.25	6.25	10.21	79.72	10.54
2	2.22	8.47	10.49	61.50	2.73
3	1.25	9.72	10.63	46.53	1.84
4	0.35	10.07	10.66	44.66	1.75

Table 6.3: LCT Algorithm, Scenario 1 - Box Plot Data

	capacities for active node pairs (in bandwidth units)		
no. of topologies	mean	variance	aggregate
1	14.88	11.93	3125.04
2	12.65	8.01	2657.46
3	12.42	6.31	2608.95
4	12.5	5.55	2624.88

Table 6.4: LCT Algorithm, Scenario 1 - Additional Data

6.3.3 Comparison of Algorithms

The performance of the LMT and LCT algorithms in scenario 1 with three output routing topologies is compared in Fig. 6.9 and Fig. 6.10. The figures also show the equivalent performance of the two algorithms when the output set of routing topologies contains only the base (network) topology, which is also the performance of the shortest path (SP) algorithm. The numerical values of the observed metrics are given in Tables 6.5 and 6.6. The network is fully utilised.

Fig. 6.9 shows box plots of the bandwidth values allocated to the active node pairs, depending on the algorithm used. The numbers in brackets on the y-axis indicate the number of routing topologies used by each algorithm. Any allocated bandwidth values above 20 units are excluded. The figure shows a reduction of the median range by 79% with the LMT algorithm and three output routing topologies (LMT(3)), and by 82% with the LCT algorithm and three output routing topologies (LCT(3)), when compared to the shortest path algorithm and the base topology only (SP(1)). So the LCT algorithm with three routing topologies has achieved the minimum variations in capacities allocated to at least half of the active node pairs.

The minimum capacities allocated to the active node pairs depending on the algorithm used are shown in Fig. 6.6, with the number of routing topologies used by each algorithm given in brackets on the x-axis. When compared to the shortest path algorithm on the

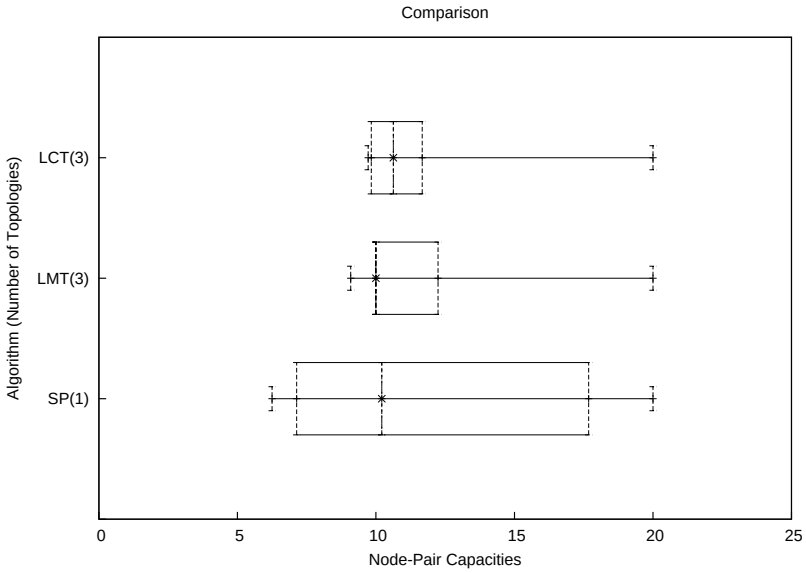


Figure 6.9: Scenario 1: Allocated Capacities

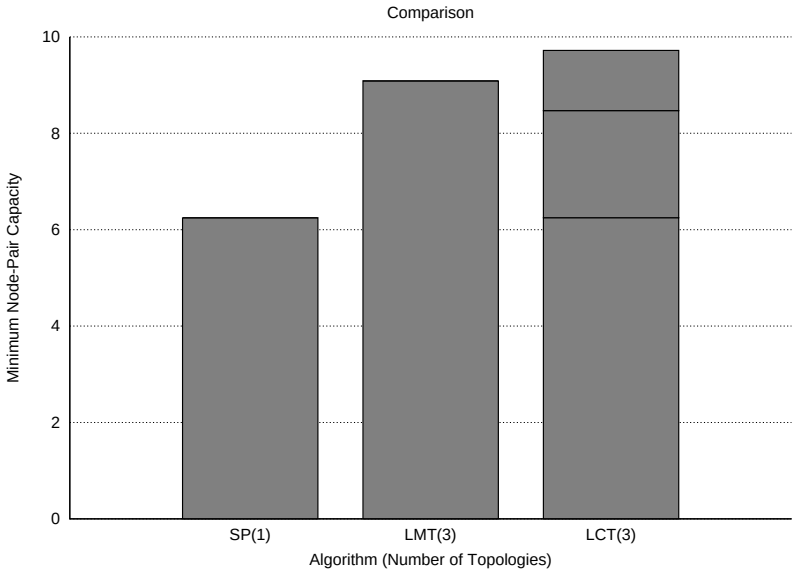


Figure 6.10: Scenario 1: Minimum Capacity

base topology (SP(1)), there is an increase of this minimum capacity by 45% when the LMT algorithm with three output routing topologies is used, and by 56% when the LCT algorithm with three output routing topologies is used.

Overall, in this scenario, the LCT algorithm has shown the best performance. The LCT algorithm has allocated capacities for an active node pair along three paths in the three routing topologies it has created. The minimum increment in capacities in these three routing topologies of 6.25, 2.22, and 1.25 units is indicated in Fig. 6.6. The capacities allocated by the LMT algorithm in its three output routing topologies, and by the

	capacities for active node pairs (in bandwidth units)			
algorithm(no. of topologies)	minimum	median	maximum	median range
SP(1)	6.25	10.21	79.72	10.54
LMT(3)	9.09	10	69.62	2.24
LCT(3)	9.72	10.63	46.53	1.84

Table 6.5: LMT vs. LCT, Scenario 1: Box Plot Data

	capacities for active node pairs (in bandwidth units)		
algorithm(no. of topologies)	mean	variance	aggregate
SP(1)	14.88	11.93	3125.04
LMT(3)	13.22	8.85	2777.11
LCT(3)	12.42	6.31	2608.95

Table 6.6: LMT vs. LCT, Scenario 1: Additional Data

shortest path algorithm in its single routing topology lie on a single path.

6.3.4 A Modified Scenario

Additional results are presented below for a modified scenario with two changes.

- The target network topology is that in Fig. 6.2 (the ISP topology). Enumeration of nodes is shown in Fig. 6.11.
- The LMT algorithm is used, but omits step 5 of the algorithm, so that the capacities remaining after the allocation in the previous four steps are not allocated using a prioritized max-min allocation (the algorithm steps are listed on page 83).

Omitting the final step of the LMT algorithm should increase its fairness, but reduce the overall level of link utilisation.

The metrics obtained below are:

- the average utilisation of network links,
- the variance of utilisation of network links, and
- the capacity allocated for every active node pair.

The results obtained are shown in Table 6.7. The capacity allocated to every active node pair depending on the number of output routing topologies provided by the LMT algorithm is also shown graphically in Fig. 6.12. The average link utilisation and the variance

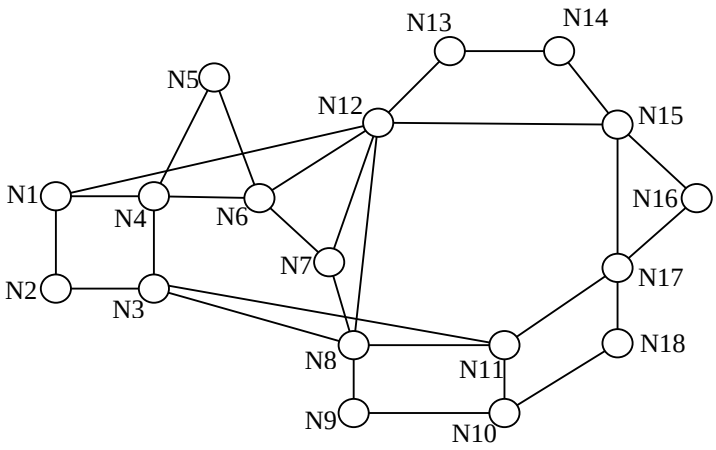


Figure 6.11: A Modified Scenario

of link utilisations as a function of the number of output routing topologies are shown in Fig. 6.13.

In comparison to the results obtained with a single output routing topology, when the performance of the LMT algorithm is equivalent to the performance of the shortest path algorithm, there is an increase in the capacity allocated to every pair of network nodes by 34% with two additional routing topologies, while the average link utilisation has been increased by 45%. As the variance in link utilisation decreases by 32%, a more even traffic distribution across network links is achieved. Additional routing topologies have not led to a further increase in allocated capacities.

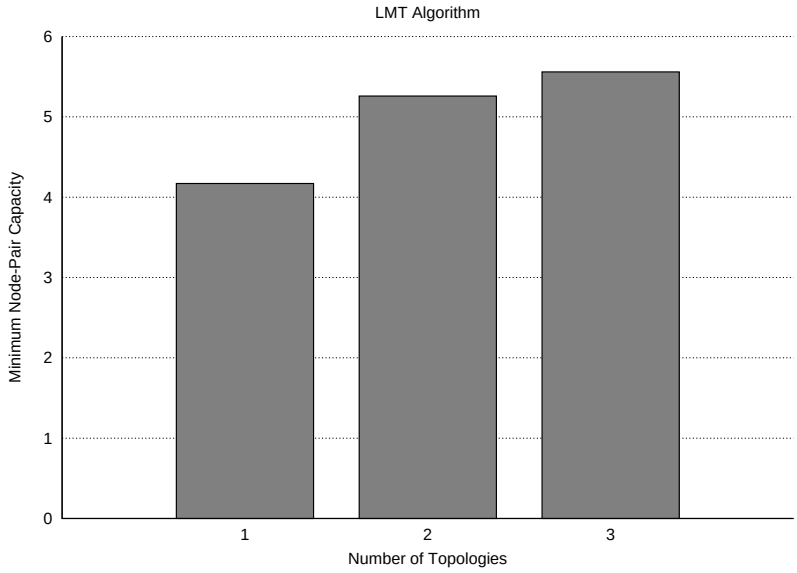


Figure 6.12: Modified Scenario: Allocated Capacity

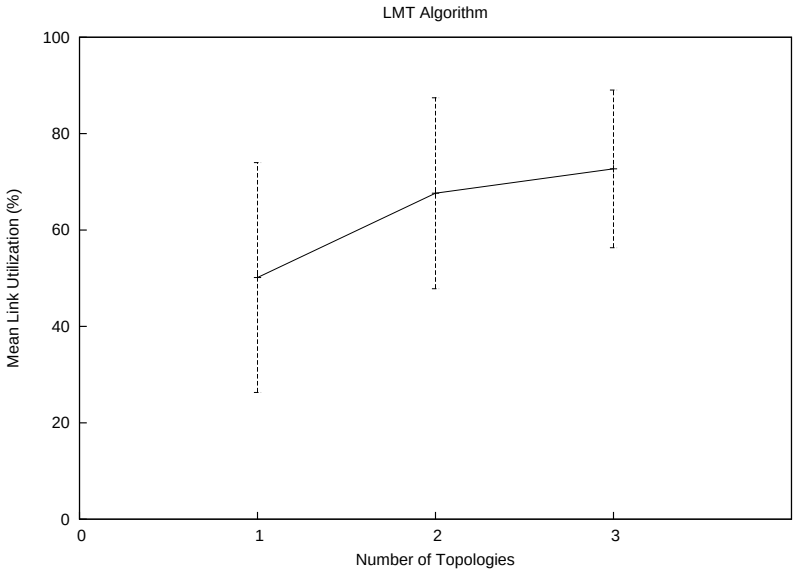


Figure 6.13: Modified Scenario: Mean (+- Variance) of Link Utilisation

	allocated capacity	link utilisation	
no. of topologies	(in bandwidth units)	mean	variance
1	4.17	50.14	23.85
2	5.26	67.63	19.81
3	5.56	72.69	16.36

Table 6.7: LMT Algorithm, Modified Scenario

Therefore, with two additional routing topologies the modified LMT algorithm has achieved an increase of network throughput by 34% by using up 45% more of network capacities. The distribution of capacities to the active node pairs is fair, but in the best case in this example nearly 30% of network capacities are not used.

6.4 Scenario 2: Unbalanced Load and Network

The second set of test results that are presented in this chapter is obtained by providing the following input information to the routing topology algorithms.

Network Topology: The input network topology is shown in Fig. 6.14. The shown topology consists of $n = 15$ nodes and $m = 28$ bidirectional edges. The set of edges, E, is divided in two subsets. They are the subset of high capacity edges $E^h = \{E_{3,6}, E_{3,7}, E_{6,3}, E_{6,10}, E_{6,11}, E_{7,3}, E_{7,10}, E_{10,6}, E_{10,7}, E_{10,11}, E_{11,6}, E_{11,10}\}$, and the subset of low capacity edges $E^l = \{E \setminus E^h\}$. The capacity of high capacity edges is

$C_{E^h} = 400$ bandwidth units. The capacity of each low capacity edge is $C_{E^l} = 100$ bandwidth units. The high capacity edges are drawn using a bold line in Fig. 6.14.

Traffic Demands: In the set of nodes, N, a subset of high demand nodes is defined, $N^h = \{N_1, N_2, N_4, N_5, N_9, N_{12}, N_{15}\}$. Every pair of high demand nodes exchange four times more traffic than is exchanged between any other two network nodes. So the estimated traffic demands $d^h_{i,j}$ for any pair of nodes $N_i \in N^h$, $N_{j,i\neq j} \in N^h$ are $d^h_{i,j} = 4k$, while the traffic demands $d^l_{i,j}$ for any other pair of nodes N_i and $N_{j,i\neq j}$ in N are $d^l_{i,j} = k$, where k is a positive value. The high demand nodes are filled in grey in Fig. 6.14.

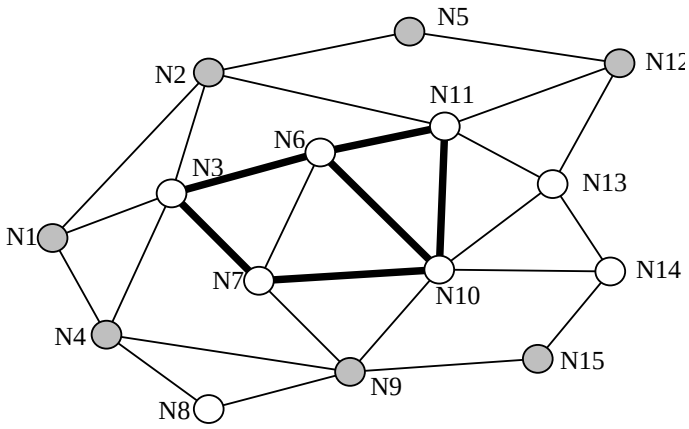


Figure 6.14: Network Topology, Scenario 2

The two configurable parameters of the LMT algorithms are set to identical values as in the first scenario: $\Delta h_{max} = 3$, $\eta = 75\%$ (section 6.3).

6.4.1 The LMT Algorithm

The bandwidth values allocated by the LMT algorithm to the pairs of nodes with low demands in scenario 2 are presented graphically in Fig. 6.15. The bandwidth values allocated to the pairs of nodes with high demands are presented in Fig. 6.16. For clearer box plot presentation, any allocated bandwidth values above 40 units are excluded from the two figures. The minimum capacities allocated by the LMT algorithm are shown in Fig. 6.17. The numerical values of the observed metrics are given in Tables 6.1 and 6.2.

The results show the following. When compared to using only a single output routing topology, an increase in the minimum allocated capacity by 24% has been achieved with

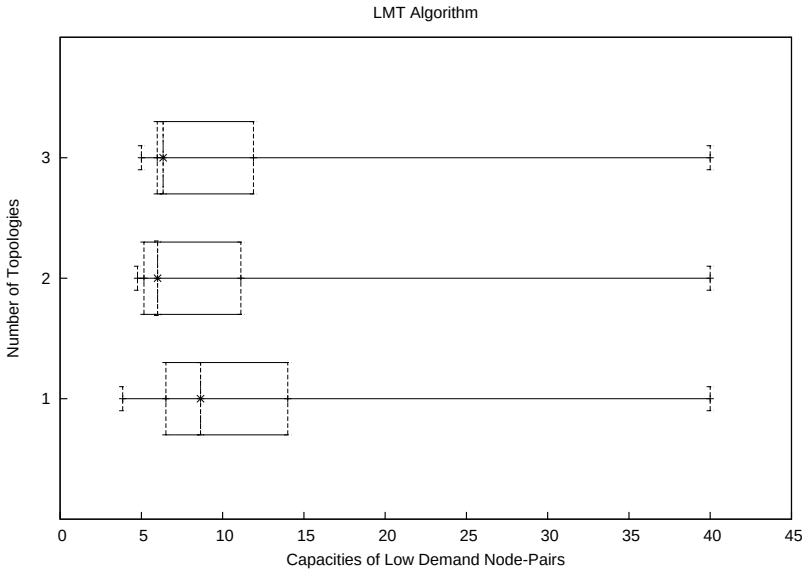


Figure 6.15: LMT Algorithm, Scenario 2: Allocated Capacities, Low Demand Pairs

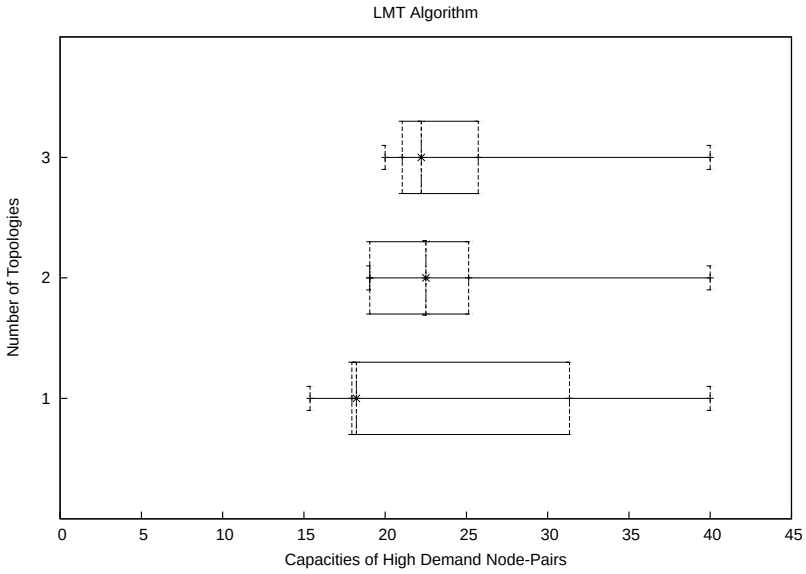


Figure 6.16: LMT Algorithm, Scenario 2: Allocated Capacities, High Demand Pairs

one additional routing topology defined by the LMT algorithm, and by 30% with two additional routing topologies. No further increase of the minimum allocated capacity was obtained with more output routing topologies. The variations in bandwidth values allocated to at least half of the node pairs with high demands are reduced by 55% and by 65% in the two cases, respectively. The variations in bandwidth values allocated to at least half of node pairs with low demands are reduced by approximately 21%. Therefore, the asymmetries introduced in scenario 2, when compared to scenario 1, have not adversely affected the performance of the LMT algorithm.

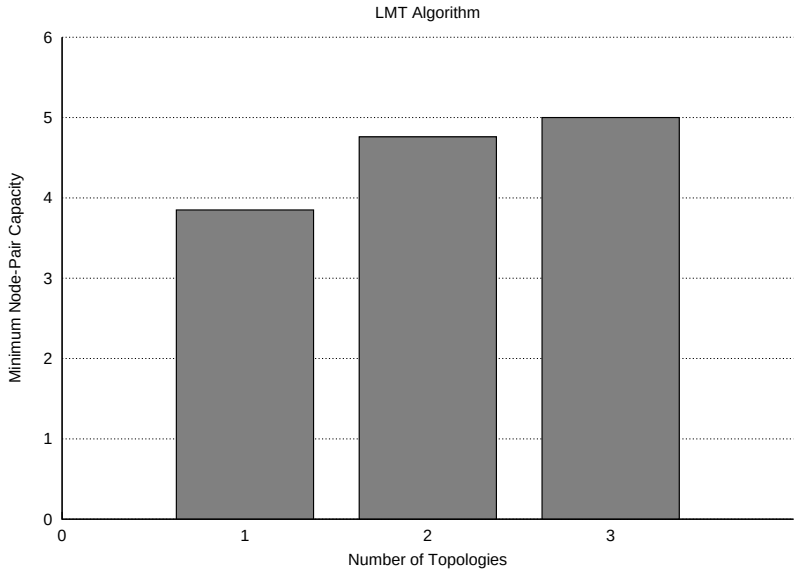


Figure 6.17: LMT Algorithm, Scenario 2: Minimum Capacity

	capacities for active node pairs (in bandwidth units)			
no. of topologies	minimum	median	maximum	median range
1	3.85	8.64	273.97	7.50
2	4.76	6.00	254.74	5.96
3	5	6.34	266.62	5.91

Table 6.8: LMT Algorithm, Scenario 2, Low Demand Node Pairs

	capacities for active node pairs (in bandwidth units)			
no. of topologies	minimum	median	maximum	median range
1	15.38	18.23	45.87	13.39
2	19.05	22.51	44.27	6.09
3	20	22.22	31.58	4.68

Table 6.9: LMT Algorithm, Scenario 2, High Demand Node Pairs

6.4.2 The LCT Algorithm

The bandwidth values allocated by the LCT algorithm to the pairs of nodes with low demands in scenario 2 are shown in Fig. 6.18, and the bandwidth values allocated to the pairs of nodes with high demands are shown in Fig. 6.19. For clearer box plot presentation any allocated bandwidth values above 40 units are excluded from the two figures. The minimum capacities allocated by the LCT algorithm are shown in Fig. 6.20. The numerical values of the observed metrics are given in Table 6.1 and Table 6.2.

The performance of the LCT algorithm in scenario 2 does not significantly differ from that in scenario 1. The LCT algorithm has increased the minimum capacity by 25%

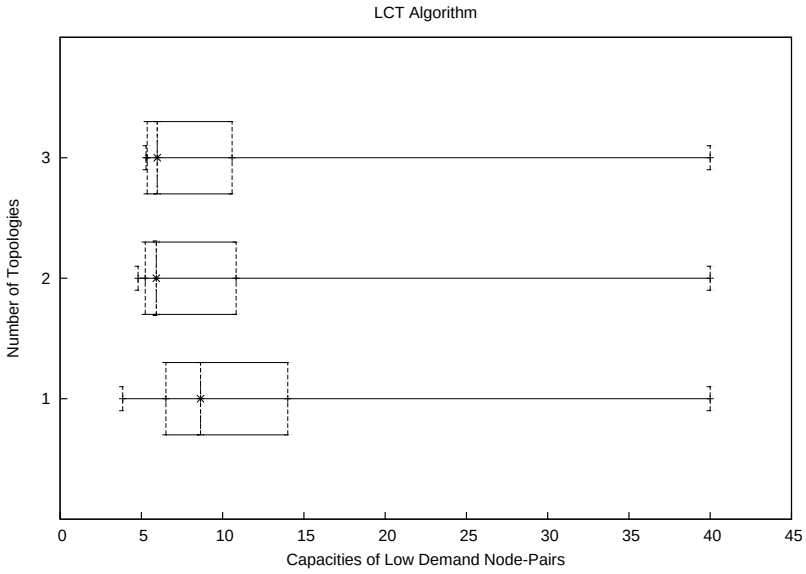


Figure 6.18: LCT Algorithm, Scenario 2: Allocated Capacities, Low Demand Pairs

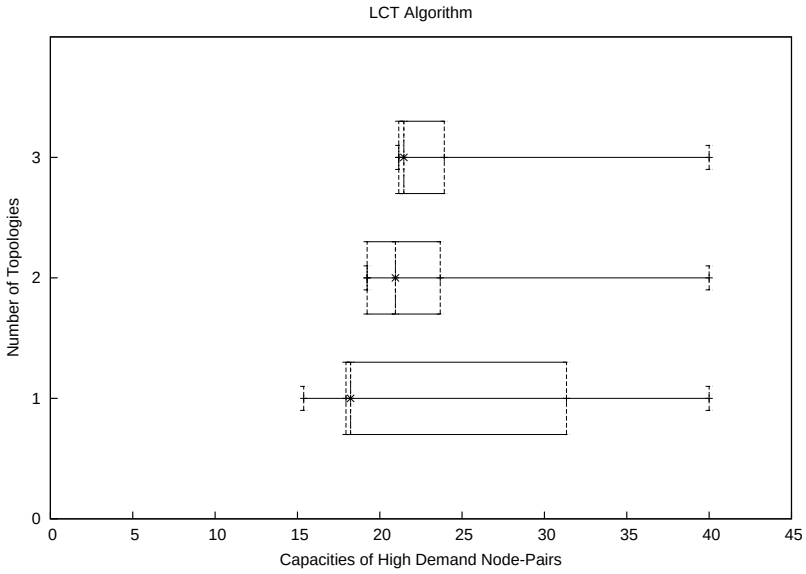


Figure 6.19: LCT Algorithm, Scenario 2: Allocated Capacities, High Demand Pairs

with one additional routing topology, and by 37% with two additional routing topologies defined by the algorithm, compared to the case with a single output routing topology. No further increase of the minimum capacity was obtained with more output routing topologies. The variations in bandwidth values allocated to at least half of the node pairs with high demands are reduced by 67% and by 79% in the two cases, respectively. The variations in bandwidth values allocated to at least half of the node pairs with low demands are reduced by 25% and by 31% in the two cases, respectively.

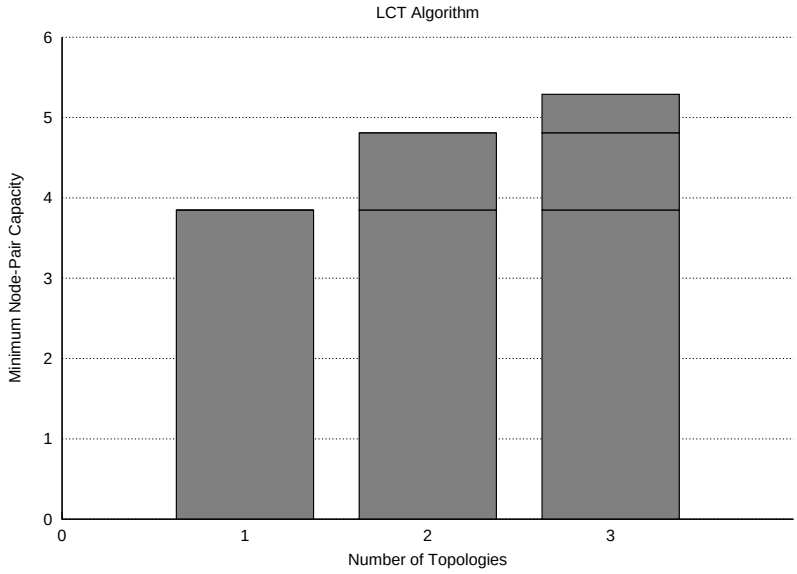


Figure 6.20: LCT Algorithm, Scenario 2: Minimum Capacity

	capacities for active node pairs (in bandwidth units)				
no. of topologies	tpl. minimum	minimum	median	maximum	median range
1	3.85	3.85	8.64	273.97	7.50
2	0.96	4.81	5.92	261.26	5.59
3	0.48	5.29	5.98	259.17	5.21

Table 6.10: LCT Algorithm, Scenario 2, Low Demand Node Pairs

	capacities for active node pairs (in bandwidth units)			
no. of topologies	minimum	median	maximum	median range
1	15.38	18.23	45.87	13.39
2	19.23	20.96	53.85	4.44
3	21.15	21.46	46.15	2.79

Table 6.11: LCT Algorithm, Scenario 2, High Demand Node Pairs

6.4.3 Comparison of Algorithms

The performance of the LMT and LCT algorithms in scenario 2 with three output routing topologies is compared in Fig. 6.21, Fig. 6.22 and Fig. 6.23. The figures also show the performance of the shortest path (SP) algorithm. The bandwidth values allocated to the pairs of nodes with low demands, and high demands, are shown in Fig. 6.21 and Fig. 6.22, respectively. For clearer presentation, allocated bandwidth values above 40 units are excluded in box plots. The box plot data is given in Tables 6.12 and 6.13. The network is fully utilised.

The results show an increase of the minimum capacity by 30% when the LMT al-

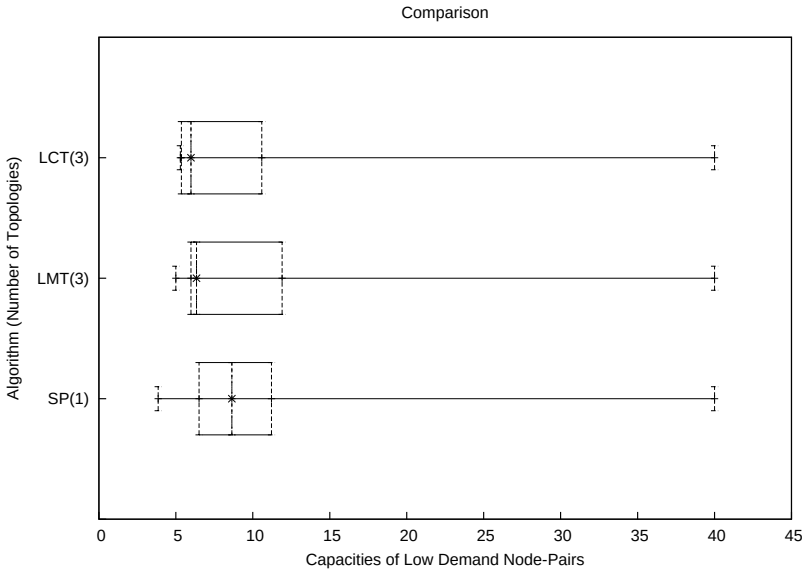


Figure 6.21: Scenario 2, Comparison: Low Demand Pairs

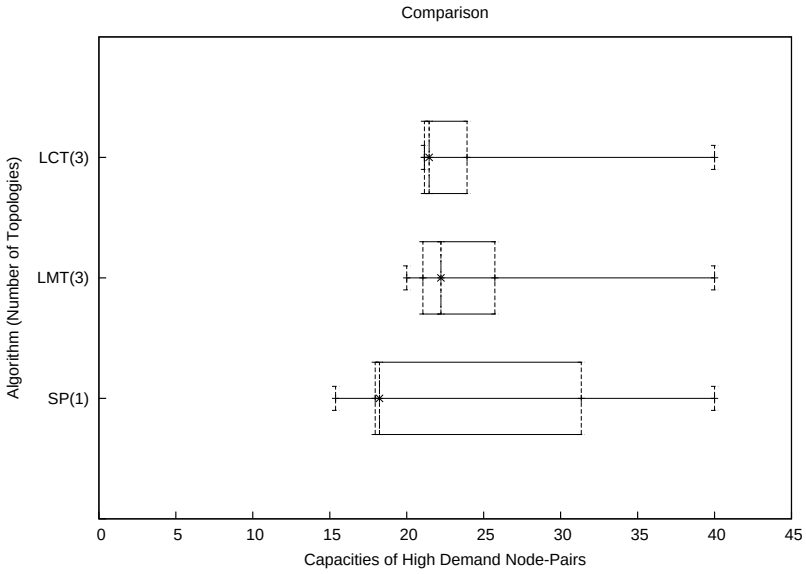


Figure 6.22: Scenario 2, Comparison: High Demand Pairs

gorithm with three output routing topologies is used (LMT(3)), and by 37% when the LCT algorithm with three output routing topologies is used (LCT(3)), in comparison to the shortest path algorithm and the base topology only (SP(1)). The LCT algorithm with three routing topologies has achieved the minimum variations in capacities allocated to at least half of the node pairs with high demands, 79% lower than the variations achieved with the shortest path algorithm, and it has achieved the minimum variations in the capacities allocated to at least half of the node pairs with low demands, which are 31% lower than the variations achieved with the shortest path algorithm.

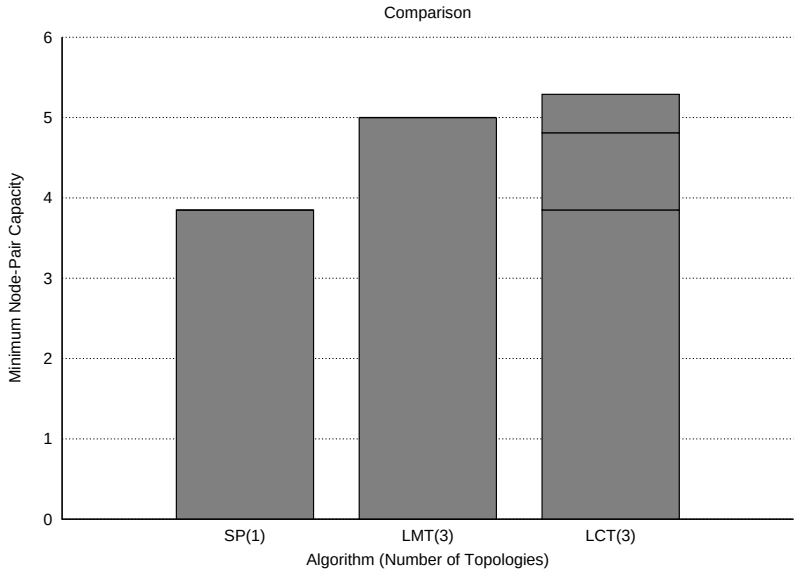


Figure 6.23: Scenario 2, Comparison: Minimum Capacities

	capacities for active node pairs (in bandwidth units)			
algorithm(no. of topologies)	minimum	median	maximum	median range
SP(1)	3.85	8.64	273.97	7.5
LMT(3)	5	6.34	266.62	5.9
LCT(3)	5.28	5.98	259.17	5.22

Table 6.12: LMT vs. LCT, Scenario 2: Box Plot Data, Low Demand Node Pairs

	capacities for active node pairs (in bandwidth units)			
algorithm(no. of topologies)	minimum	median	maximum	median range
SP(1)	15.38	18.23	45.87	13.39
LMT(3)	20	22.22	31.58	4.68
LCT(3)	21.15	21.46	46.15	2.77

Table 6.13: LMT vs. LCT, Scenario 2: Box Plot Data, High Demand Node Pairs

Overall, in the observed scenario, the LCT algorithm has shown the best performance.

6.5 Summary

The performance evaluation of the LMT algorithm and the LCT algorithm has been undertaken in this chapter. The presented test results show that there are cases when the analyzed routing topology algorithms can significantly (by 30%-60%) and fairly increase the throughput of networks compared to conventional shortest path routing with only a few additional routing topologies. Such improvements are obtained on both balanced and

unbalanced networks. The performance of the routing algorithms is a function of:

- the input network topology,
- the input traffic demands, and
- the definition of shortest paths.

The additional overhead incurred in deploying these routing schemes is modest when the number of routing topologies is low, and the improvements in network performance they produced in the scenarios investigated here are significant. This suggest that their deployment in real network will be beneficial.

Chapter 7

Conclusion

As the Internet is taking an increasingly central role in our communications infrastructure and network providers are struggling to provide sufficient network resources for the constantly growing demands, the importance of ensuring that network resources are fully utilised is growing. There is an extensive ongoing research on advanced traffic engineering techniques and efficient Quality of Service (QoS) routing solutions. In this thesis traffic control options in IP and MPLS networks were reviewed, and a need for more flexible solutions was identified. Modifications of network protocols were advocated and two novel traffic control approaches were proposed and evaluated.

7.1 Overview

A general framework to describe routing solutions for packet-switched networks was presented. Intra-area routing solutions in IP and MPLS networks were then described in the context of this framework. The primary disadvantages of both routing solutions were subsequently identified as the following:

- the lack of options for the establishment of forwarding topologies, and the use of routing topologies (as defined within the general framework),
- the lack of options for the establishment and modification of forwarding layers, and
- the inability of the forwarding protocol to forward packets along any type of forwarding topology, regardless of their priority or TOS.

The presence of point-to-point forwarding layers is crucial for providing flexibility in controlling traffic routes, whereas the presence of other types of forwarding layers and use of routing topologies are crucial for reducing the information overhead and for simplifying the network management.

These limitations motivated the design of a new routing solution. The most important novelties in that new solution are:

- the control protocol with mechanisms for controlling factors that affect traffic routes, including forwarding topologies, routing topologies, and routing assignments, and
- the forwarding protocol which adds a field prefix to the forwarding header of a packet when the packet starts following a forwarding topology (like in MPLS). This prefix contains the unique short identifier of that forwarding topology, which (unlike in MPLS) is centrally defined for all nodes in that topology, and which enables forwarding of the packet via that forwarding topology.

The use of this short prefix, like in MPLS, is important for reducing the amount of header analysis in the forwarding process when compared to the IP solution. The use of short identifiers is important for speeding up table lookups when compared to the IP solution where for the same purpose variable length address prefixes are used. Both features are important for gaining faster forwarding speeds. By using these centrally defined identifiers of forwarding topologies, and not identifiers defined independently for each network node as in MPLS, the need for a separate label binding and distribution scheme depending on the type of forwarding and routing topology used is eliminated. This is important for simplifying and speeding up network protocol operation. Like explicit MPLS routing, the new routing solution recommended in this thesis would have as an additional advantage over hop-by-hop IP routing better traffic engineering capabilities due to its application of point-to-point forwarding layers, and so its better handling of quality of service issues.

The most common choice of a traffic route in current networks is the shortest path. Sometimes congestion may be experienced on the shortest paths as they have insufficient capacities for the current traffic demands, which could be avoided if alternative paths were

used instead. The network resources are then used inefficiently, as the network handles less traffic than it could. For increasing the network throughput algorithms are needed which would discover and direct traffic to the alternative paths when beneficial. Two such algorithms were introduced in this thesis. They are called the Link Mask Topology (LMT) algorithm and the Link Cost Topology (LMT) algorithms. These algorithms are routing topology algorithms as they discover alternative paths by creating a set of routing topologies, based on the given network topology and demands. The shortest paths in these routing topologies are to be used as traffic routes. However, as the output routing topologies may differ from the base network topology in the number of edges or the edge lengths, the shortest paths they provide are not necessarily the same as shortest paths through the network. This allows traffic to be balanced across the network. The following are the major difference between the two algorithms.

- The LMT algorithm directs **all** traffic on the shortest network paths between **some** active node pairs (i.e. those with non-zero traffic demands) to alternative routes.
- The LCT algorithm directs **some** traffic on the shortest paths network between **all** active node pairs to alternative routes.

Therefore, the LCT algorithm relies on the presence of traffic tuning techniques in the network which can split the traffic to be carried between two network nodes onto multiple selected paths in the ratio defined by this algorithm. The LMT algorithm has no such requirements, as it directs traffic between any two network nodes to a single path.

The performance of the LMT algorithm and the LCT algorithm has been evaluated in the thesis in a number of scenarios and compared to the shortest path algorithm. The key performance objective of these algorithms is to maximize network throughput while maintaining fairness in the allocation of network resources. The test results show the potential of both proposed algorithms to significantly increase the minimum capacity assigned to an active node pair, and the network throughput, by 30%-60% (with only two additional routing topologies), when compared to the shortest path routing algorithm. The variations between the capacities assigned to at least half of active node pairs are substantially reduced too, by more than 70%. They thus constitute a viable approach to the provision of flexible and efficient traffic control solutions for packet-switched networks.

7.2 Future Work

The routing tasks which have not been studied in this thesis include traffic state monitoring, resource management and the prioritization of packets, and will be addressed in future work. Other existing routing solutions need to be examined too. Low level requirements are then to be defined and potential problems in their realization to be considered. The performance of the two routing topology algorithms introduced in this thesis will be compared to other algorithms with similar performance objectives in real networks, such as techniques for constraint-based routing.

Bibliography

[1] D.O.Awduche, A. Chiu, A.Elwaid, I.Widjaja, and X.Xiao. RFC 3272: Overview and Principles of Internet Traffic Engineering. *Status: Informational*, May 2002.

[2] K. Nichols, S. Blake, F. Baker, and D. Black. RFC 2386: A Framework for QoS-based Routing in the Internet. *Status: Informational*, August 1998.

[3] X. Xiao and L. M. Ni. Internet QoS: A big picture. *IEEE Network*, 13(2):8–18, 1999.

[4] J. Tian Y. Zheng, W. Dou and M. Xiao. An Overview of research on QoS routing. *Advanced Parallel Processing Technologies*, pages 387–397, 2003.

[5] J. Domingo-Pascual A. Fonte M. Curado E. Monteiro F. Kuipers P. Van Mieghem S. Avallone G. Ventre P. Arana-Gutierrez M. Hollick R. Steinmetz L. Iannone X. Masip-Bruin, M. Yannuzzi and K. Salamatian. Research challenges in QoS routing. *Elsevier Journal of Computer Communications*, pages 563–581, 2005.

[6] R. Guerin G. Apostolopoulos and S. Kamat. Implementation and performance measurements of QoS routing extensions to OSPF. *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 2:680–688, March 1999.

[7] J. Moy. RFC 2328: OSPF Version 2. *Status: Standards Track*, April 1998.

[8] D. Oran. RFC 1142: OSI IS-IS Intra-domain Routing Protocol. *Status: Informational*, February 1990.

[9] Cisco Systems. Configuring OSPF. *http://www.cisco.com/univerc/cc/td/doc/product/software/ios113ed/113ed cr/np1 c/1cospf.htm*, 1997.

[10] V. Fuller, T. Li, J. Yu, and K. Varadhan. RFC 1519: Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy. *Status: Proposed Standard*, September 1993.

[11] K. Lougheed and Y. Rekhter. RFC 1163: A Border Gateway Protocol (BGP). *Status: Proposed Standard*, June 1990.

[12] DARPA. RFC 791: Internet Protocol. *Status: Standard*, September 1981.

[13] J. Mogul and S. Deering. RFC 1191: Path MTU Discovery. November 1990.

[14] J. Postel. RFC 792: Internet Control Message Protocol. *Status: Standard*, September 1981.

[15] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik. 1*, pages 269–271, 1959.

[16] N. Santoro M. D. Atkinson, J.-R. Sack and T. Strothotte. Min-max heaps and generalized priority queues. *Programming techniques and Data structures. Communications of ACM*, 29(10):996–1000, October 1996.

[17] M. L. Fredman and R. E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM 34(3)*, pages 596–615, 1987.

[18] R. H. Mhring E. Khler and H. Schilling. Acceleration of Shortest Path and Constrained Shortest Path Computation. *Experimental and Efficient Algorithms*, 3530/2005:126–138, May 2005.

[19] H. Kaplan A. V. Goldberg and R. F. Werneck. Efficient point-to-point shortest path algorithms. *In Proceedings of the 8th Workshop on Algorithm Engineering and Experiments (ALENEX'06)*, pages 129–143, 2006.

[20] B. Schtz D. Wagner R.H. Mhring, H. Schilling and T. Willhalm. Partitioning graphs to speedup Dijkstra's algorithm. *Journal of Experimental Algorithmics (JEA), Section 2 - Selected papers from WEA 2005*, 11(Article No. 2.8), October 2007.

[21] P. Sanders D. Schieferdecker D. Schultes R. Bauer, D. Delling and D. Wagner. Combining hierarchical and goal-directed speed-up techniques for Dijkstra's algorithm. *In Proceedings of the 7th Workshop on Experimental Algorithms (WEA'08)*, pages 303–318, May 2008.

[22] R. Bauer and D. Delling. Fast and robust unidirectional routing. *Journal of Experimental Algorithmics (JEA), Section 2 - Selected Papers from ALENEX 2008*, 14(Article No. 4), December 2009.

[23] J. Turnerz M. Waldvogely, G. Varghesez and B. Plattnery. Scalable high speed IP routing lookups. *ACM SIGCOMM Computer Communication Review*, 27(4):25–36, October 1997.

[24] E. W. Biersack M. A. Ruiz-Snchez and W. Dabbous. Survey and taxonomy of IP address lookup algorithms. *IEEE Network*, 15(2):8–23, January 2001.

[25] K. Ramakrishnan and S. Floyd. RFC 3168: The Addition of Explicit Congestion Notification (ECN) to IP. *Status: Standards Track*, September 2001.

[26] S. Floyd. RFC 2914: Congestion Control Principles. *Status: Best Current Practice*, September 2000.

[27] T. Harks L. Mamatas and V. Tsaoussidis. Approaches to congestion in packet networks. *Journal of Internet Engineering*, 1(1):22–33, 2007.

[28] T. Cicic S. Gjessing A. Kvalbein, A. F. Hansen and O. Lysne. Resilient routing layers for recovery in packet networks. *Dependable Systems and Networks, 2005. DSN 2005. Proceedings.*, pages 238–247, July 2005.

[29] T. Cicic S. Gjessing O. Lysne T. Jensen A. F. Hansen, A. Kvalbein and O. N. sterb. Fast, effective and stable IP recovery using resilient routing layers. *The 19th International Teletraffic Congress (ITC19) Beijing*, pages 1631–1640, August 2005.

[30] T. Schwabe M. C. Scheffel, C. G. Gruber and R. G. Prinz. Optimal multi-topology routing for IP resilance. *AEU - International Journal of Electronics and Communications*, 60(1):35–39, January 2006.

[31] T. Cicic S. Gjessing A. Kvalbein, A. F. Hansen and O. Lysne. Fast IP network recovery using multiple routing configurations. *In Proc. INFOCOM 2006*, pages 23–29, 2006.

[32] T. Cicic. On basic properties of fault-tolerant multi-topology routing. *Computer Networks: The International Journal of Computer and Telecommunications*, 15(18):3325–3341, December 2008.

[33] M.Menth and R.Martin. Network Resilience Through Multi-topology Routing. *Design of Reliable Communication Networks, 2005 (DRCN 2005), Proceedings*, pages 271–277, October 2005.

[34] A. Roy L. Nguyen P. Psenak, S. Mirtorabi and P. Pillay-Esnault. RFC 4916: MT-OSPF: Multi Topology (MT) Routing in OSPF. *Status: Standards Track*, January 2007.

[35] N. Shen T. Przygienda and N. Sheth. RFC 5120: M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs). *Status: Standards Track*, February 2008.

[36] C. Hopps. RFC 2992: Analysis of an Equal-Cost Multi-Path Algorithm. *Status: Informational*, November 2000.

[37] DARPA. RFC 793: Transport Control Protocol. *Status: Standard*, September 1981.

[38] B. Fortz and M. Thorup. Internet traffic engineering by optimizing OSPF weights. *in Proc. IEEE INFOCOM*, pages 519–528, 2000.

[39] J. Rexford B. Fortz and M. Thorup. Traffic Engineering with Traditional IP Routing Protocols. *IEEE Communications Magazine*, pages 118–124, October 2002.

[40] A. El-Sayed A.A.Ghazala and M. Mousa. A survey for open shortest path first weight setting (OSPFWS) problem. *Proceedings of the 2008 International Conference on Information Security and Assurance (isa 2008)*, pages 111–116, 2008.

[41] A. El-Sayed A.A.Ghazala and M. Mousa. A new approach for open shortest path weight setting problem (OSPFWSP). *In Proc. of the 2008 Third International Conference on Convergence and Hybrid Information Technology*, pages 188–193, 2008.

[42] W.Ben-Ameur, N.Michel, E.Gourdin, and B.Liau. Routing strategies for IP Networks. *Telektronikk*, 2(3):145–158, 2001.

[43] M.G.C.Resende M.Ericsson and P.M.Pardalos. A genetic algorithm for the weight setting problem in OSPF routing. *Journal of Combinatorial Optimization*, pages 299–333, Novemeber 2004.

[44] R. Guerin A. Sridharan and C. Diot. Achieving near-optimal traffic engineering solutions for current OSPF/IS-IS networks. *IEEE/ACM Transactions on Networking*, 13(2):234–247, April 2005.

[45] B. Fortz and M. Thorup. Robust optimization of OSPF/IS-IS weights. *In Proc. International Network Optimization Conference*, pages 225–230, 2003.

[46] S. Bhattacharyya N. Taft A. Nucci, B. Schroeder and C. Diot. IGP Link Weight Assignment for Transient Link Failures. *in 18th International Teletraffic Congress*, August 2003.

[47] A. Sridharan and R. Guerin. Making IGP routing robust to link failures. *in Proceedings of Networking*, 2005.

[48] B. Fortz and M. Thorup. Optimizing OSPF/IS-IS weights in a changing world. *in IEEE JSAC Special Issue on Advances in Fundamentals of Network Management*, 2002.

[49] A. Basu and J. G. Riecke. Stability issues in OSPF routing. *in Proceedings of SIGCOMM*, pages 225–236, August 2001.

[50] J. Rexford A. Shaikh and K. G. Shin. Evaluating the impact of stale link state on Quality-of-Service routing. *IEEE/ACM Transactions on Networking*, 9(2):162–176, April 2001.

[51] T. Griffin R. Teixeira, A. Shaikh and G. M. Voelker. Network sensitivity to hot-potato disruptions. *in Proceedings of SIGCOMM*, pages 231–244, August 2004.

[52] L. Adamovic and M. Collier. A new traffic engineering approach for IP networks. *in proceedings of CSNDSP 2004*, June 2004.

[53] K. Kowalik L. Adamovic and M. Collier. Traffic control in IP Networks with multiple topology routing. *Networking - ICN 2005*, pages 335–342, April 2005.

[54] A. Kvalbein and O. Lysne. How can multi-topology routing be used for intradomain traffic engineering? *Applications, Technologies, Architectures, and Protocols for Computer Communication Proceedings of the 2007 SIGCOMM workshop on Internet network management*, pages 280–284, 2007.

[55] F. Baker K. Nichols, S. Blake and D. Black. RFC 2474: Deffinition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. *Status: Standards Track*, December 1998.

[56] R. Braden, D. Clark, and S. Shenker. RFC 1633: Integrated Services in the Internet Architecture: an Overview. *Status: Proposed Standard*, June 1994.

[57] M. Atiquzzaman H. Bai and W. Ivancic. Running Integrated Services over Differentiated Services Networks: Quatitative Performance Measurements. *in proceedings of SPIE, Quality of Service over Next-Generation Internet*, 4866:11–22, August 2002.

[58] F. Baumgarter R. Balmer and T. Braun. A Concept for RSVP over Diffserv. *in proceedings of the 9th International Conference on Computer Communication and Network, Las Vegas*, pages 412–417, October 2002.

[59] R. Yavatkar F. Baker L. Zhang M. Speer R. Braden B. Davie J. Wroclawski Y. Berent, P. Ford and E. Felstaine. RFC 2998: A Framework for Integrated Services Operation over DiffServ Networks. *Status: Informational*, November 2000.

[60] R. Callon E. Rosen, A. Viswanathan. RFC 3031: Multiprotocol Label Switching Architecture. *Status: Standards Track*, January 2001.

[61] J. Ash, M. Girish, E. Gray, B. Jamoussi, and G. Wright. RFC 3213: Applicability Statement for CR-LDP. *Status: Informational*, January 2002.

[62] D. Awduche, A. Hannan, and X. Xiao. RFC 3210: Applicability Statement for Extensions to RSVP for LSP-Tunnels. *Status: Informational*, December 2001.

[63] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus. RFC 2702: Requirements for Traffic Engineering Over MPLS. *Status: Informational*, September 1999.

[64] S. Shenker, C. Partridge, and R. Guerin. RFC 2212: Specification of Guaranteed Quality of Service. *Status: Standards Track*, September 1997.

[65] J. Wroclawski. RFC 2211: Specification of the Controlled Load Network Element Service. *Status: Proposed Standard*, September 1997.

[66] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. RFC 2205: Resource ReSerVation Protocol (RSVP). *Status: Standards Track*, September 1997.

[67] D. Mitra E. Bouillet and K.G. Ramakrishnan. The structure and management of service level agreements in networks. *Selected Areas in Communications, IEEE Journal*, 20(4):691–699, May 2002.

[68] D.C. Verma. Service level agreements on IP networks. *Proceedings of the IEEE*, 92(9):1382–1388, Septemeber 2004.

[69] W. Weiss J. Heinhanen, F. Baker and J. Wroclawski. RFC 2597: Assured Forwarding PHB Group. *Status: Standards Track*, June 1999.

[70] V. Jacobson K. Nichols and L. Zhang. RFC 2638: A Two-bit Differentiated Services Architecture for the Internet. *Status: Informational*, July 1999.

[71] Y. Cheng and W. Zhuang. Dynamic inter-SLA resource sharing in path-oriented differentiated services networks. *IEEE/ACM Transactions on Networking*, 14(3):657–670, June 2006.

[72] S. Tong and O.W.W. Yang. Bandwidth Management for Supporting Differentiated Service Aware Traffic Engineering. *IEEE/ACM Transactions on Parallel and Distributed Systems*, 18(9):1320–1331, September 2007.

[73] C. Tham and T. C. Hui. Reinforcement learning-based dynamic bandwidth provisioning for quality of service in differentiated services networks. *Computer Communications*, 28(15):1741–1751, September 2005.

[74] X. Zhou J. Wei, C. Xu and Q. Li. A robust packet scheduling algorithm for proportional delay differentiation services. *Computer Communications*, 29(18):3679–3690, Novemeber 2006.

[75] D. Ippoliti X. Zhou and L. Zhang. Fair bandwidth sharing and delay differentiation: Joint packet scheduling with buffer management. *Computer Communications*, 31(17):4072–4080, Novemeber 2008.

[76] A. E. Kamal and S. Sankaran. A combined delay and throughput proportional scheduling scheme for differentiated services. *Computer Communications*, 29(10):1754–1771, June 2006.

[77] G. P. Chandranmenon and G. Varghese. Trading packet headers for packet processing. *IEEE/ACM Transactions on Networking*, 4(2):141–152, 1996.

[78] L. Ni X. Xiao and V. Vuppala. A Comprehensive Comparison of IP Switching and Tag Switching. *Proceedings of IEEE Intl. Conf. on Parallel and Distributed Systems (ICPADS '97)*, pages 669–675, December 1997.

[79] David E. McDysan and Darren L. Spohn. *ATM Theory and Applications*. McGraw-Hill, September 1998.

[80] A. Pzygienda S. Kamat R. Guerin, D. Williams and A. Orda. RFC 2676: QoS Routing Mechanisms and OSPF Extentions. *Status: Experimental*, December 1998.

[81] D. Katz and D. Yeung. RFC 3630: Traffic Engineering Extentions to OSPF. *Status: Standards Track*, September 2003.

[82] H. Smit and T. Li. RFC 5305: IS-IS Extentions for Traffic Engineering. *Status: Standards Track*, October 2008.

[83] M. Krunz F. A. Kuipers, T. Korkmaz and P. Van Mieghem. A review of Constraint-Based routing algorithms. *Technical Report, can be found at http://www.nas.its.tudelft.nl/people/Fernando/papers/TRreviewqosalg.pdf*, June 2002.

[84] Y. Xu and G. Zhang. Models and algorithms of QoS-based routing with MPLS traffic engineering. *5th IEEE International Conference on High Speed Networks and Multimedia Communications*, pages 128–132, November 2002.

[85] J.K. Muppala Y. Yang and S.T. Chanson. Quality of service routing algorithms for bandwidth-delay constrained applications. *Ninth International Conference on Network Protocols*, pages 62–70, Novemeber 2001.

[86] L.S.N. Fonseca G.B. Figueiredo and J.A.S. Monteiro. A minimum interference routing algorithm with reduced computational complexity. *Computer Networks*, 10(11):1710–1732, August 2006.

[87] M. Kodialam K. Kar and T. V. Lakshman. Minimum interference routing of bandwidth guaranteed tunnels with MPLS traffic engineering applications. *IEEE Journal on Selected Areas in Communications*, 18(12):2566–2579, December 2000.

[88] M. S. Kodialam and T. V. Lakshman. Minimum interference routing with applications to MPLS traffic engineering. *INFOCOM (2)*, pages 884–893, December 2000.

[89] T.V. Lakshman M. Kodialam and S. Sengupta. A simple traffic independent scheme for enabling restoration oblivious routing of resilient connections. *INFOCOM 2004 Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 884–893, January 2004.

[90] D. Bauer S. Suri, M. Waldvogel and P. R. Warkhede. Profile-Based routing and traffic engineering. *Computer Communications*, pages 351–365, March 2003.

[91] G. Swallow P. Pan and A. Atlas. RFC 4090: Fast Reroute Extentions to RSVP-TE for LSP Tunnels. *Status: Standards Track*, May 2005.

[92] D. Applegate and M. Thorup. Load optimal MPLS routing with N+M labels. *in Proc. IEEE INFOCOM 2003*, pages 555–565, 2003.

[93] S. Ganguly S. Bhatnagar and B. Nath. Creating multipoint-to-point LSPs for traffic engineering. *IEEE Communications Magazine*, 43(1):95–100, April 2005.

[94] A. Kumar A. Gupta and R. Rastogi. Traveling with a pez dispenser (or, routing issues in MPLS). *SIAM Journal on Computing*, 34(2):453–474, 2005.

[95] H. Saito, Y. Miyao, and M. Yoshida. Traffic engineering using multiple multipoint-to-point LSPs. *in Proc. IEEE INFOCOM'2000*, pages 894–901, 2000.

[96] R. Fabregat F. Solano and J. L. Marzo. Full label space reduction in MPLS networks: asymmetric merged tunneling. *IEEE Communications Letters*, 9(11):1021–1023, November 2005.

[97] R. Fabregat F. Solano, T. Stidsen and J. L. Marzo. Label space reduction in MPLS networks: how much can a single stacked label do? *IEEE/ACM Transactions on Networking (TON)*, 16(6):1308–1320, December 2008.

[98] E. Rosen, D. Tappan, G. Fedorkow, Y. Rekhter, D. Farinacci, T. Li, and A. Conta. RFC 3032: MPLS Label Stack Encoding. *Status: Standards Track*, January 2001.

[99] T. Chen and T. Oh. Reliable Services in MPLS. *IEEE Communications Magazine*, pages 58–62, December 1999.

[100] Radia Perlman. *Interconnections: bridges, routers, switches and internetworking protocols, second edition*. Addison Wesley, September 2001.

[101] G. Armitage. MPLS: The magic behind the myths. *IEEE Communications Magazine*, pages 124–131, January 2000.

[102] M. C. Sinclair. Minimum cost topology optimisation of the COST 239 European optical network. *in Proc. Int. Conf. Artificial Neural Networks and Genetic Algorithm*, pages 26–29, 1995.

[103] C. Bouras and A. Sevasti. Analytical approach and verification of a diffserv based priority service. *High-Speed Networks and Multimedia Communications, 6th IEEE Internations Conference HSNMC 2003*, pages 11–20, 2003.

[104] Y. Huang and R. Guerin. Does Overprovisioning become more or less efficient as networks grow larger? *13th IEEE International Conference on Network Protocols (ICNP'05)*, pages 225–235, 2005.

[105] M. Menth. Capacity overprovisioning for networks with resilience requirements. *ACM SIGCOMM Computer Communication Review*, 36(4):87–98, October 2006.

[106] J. M. Faffe. Bottleneck flow control. *IEEE Transactions on Communications*, 29(7):954–962, 1981.

[107] S. Chen and K. Nahrstedt. Maxmin fair routing in connection-oriented networks. *in Proceedings of Euro-Parallel and Distributed Systems Conference (Euro-PDS '98)*, pages 163–168, 1998.

[108] B. Awerbuch and Y. Shavitt. Converging to approximated max-min flow fairnes in logarithmic time. *INFOCOM '98 Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 3:1350–1357, March/April 1998.

[109] S. Kamat A. Orda G. Apostopoulos, R. Guerin and S.K. Tripathi. Intradomain QoS routing in IP networks: A feasibility and cost/benefit analysis. *IEEE Network*, pages 42–54, 1999.

[110] X. Yuan and W. Zheng. A comparative study of Quality of Service routing schemes that tolerate imprecise state information. *In Proc. Commputer Communications and Networks*, pages 230–235, 2002.

[111] K. Kowalik and M. Collier. ALCFRA - A robust routing algorithm which can tolerate imprecise network state information. *in Proceedings of 15th ITC Specialist Seminar, Wurzburg, Germany*, July 2002.