Lehrstuhl für Kommunikationsnetze
Technische Universität München

Routing Optimization and Capacity Assignment in Multi-Service IP Networks

Anton Riedl

Vollständiger Abdruck der von der Fakultät für
Elektrotechnik und Informationstechnik der Technischen Universität München
zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs
genehmigten Dissertation.

Vorsitzende:	Univ.-Prof. Dr. rer. nat. D. Schmitt-Landsiedel
Prüfer der Dissertation:	1. Univ.-Prof. Dr.-Ing. J. Eberspächer
	2. Univ.-Prof. Dr.-Ing. R. Schehrer, Universität Dortmund

Die Dissertation wurde am 06.08.2003 bei der Technischen Universität München
eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik
am 18.12.2003 angenommen.

Acknowledgement

This dissertation was written during my time as doctorate candidate at the Institute of Communication Networks (LKN) at Munich University of Technology (TUM). Throughout the years at LKN I have always enjoyed the very encouraging, deeply rewarding, and exceptionally collegial atmosphere.

To the greatest extent this is due to the enormous commitment of Prof. Dr.-Ing. Jörg Eberspächer, the institute's head and my Ph.D. advisor. He has provided an environment where it is a pleasure to conduct research and to take on scholarly responsibilities. Furthermore, he has given me the opportunity to participate in national and international conferences and committees where I have met many interesting people. I would like to thank Prof. Eberspächer very much for his support, which has made this dissertation possible.

I would also like to thank Prof. Dr.-Ing. Rudolf Schehrer for accepting to be the second auditor of my thesis. I have met Prof. Schehrer at many occasions and I have always greatly enjoyed the discussions and conversations.

Further thanks go to my assistant colleagues at the institute whom I have not only considered my co-workers but also my friends: especially Andrea Bör, Stefan Butenweg, Josef Glasmann, and Peter Jocher as members of the Network Architecture Team at LKN for providing a platform for many technical and sometimes not-so-technical discussions, including topics across all layers and disciplines; Dominic Schupke for the exchange of experience and ideas concerning optimization and programming matters; and two former colleagues and "fellows on the road of network planning", Thomas Bauschert and Jochen Frings, for numerous discussions.

I also know that my work at LKN would not have gone as smoothly without the institute's administrative staff, whose help I have always appreciated very much. I would like to express a special "thank you" to Dr. Martin Maier, who has made life a lot easier by taking over much of the unattractive and time-consuming paper work and who, in his function as network administrator, has provided a truly outstanding network support service.

Finally, I would like to thank my wife Missy for her continual support, her patience and understanding throughout the past years.

München, August 2003 *Anton Riedl*

Abstract

This dissertation investigates routing optimization and capacity assignment in IP networks with multiple services. Both are key processes of traffic engineering and network planning, needed for cost-efficient, quality of service (QoS) aware operation of IP networks.

Routing optimization provides a means to balance the traffic load in the network with the goal to improve quality of service. The main objective of our routing optimization procedures is the minimization of the maximum link utilization in the network. Two fundamentally different strategies of routing optimization are investigated.

In the first case, optimization is based on conventional routing protocols, which implement shortest-path routing based on static link metrics. By adjusting the values of these metrics (i.e., the *link weights*), the route selection process can be influenced and, thus, the path pattern optimized. A significant novelty of our approach is the consideration of additive as well as concave link metrics. This approach allows a more flexible route selection and, therefore, provides a greater optimization potential. The link weight setting problem is formulated as a mixed-integer linear program, which for small networks can be solved with standard tools. For larger networks, a hybrid genetic algorithm is presented.

In the second case, source routing is utilized as a complementary technology to improve an existing routing configuration. In order to avoid having to establish dedicated paths between all nodes of a network, a hybrid approach is pursued. The majority of traffic is forwarded along shortest paths in native IP fashion and only a small number of flows is selected and subject to source routing. For this type of routing optimization two mixed-integer linear programs are presented. Starting with a given configuration of shortest-path routing, an optimal set of flows is identified for rerouting and the corresponding paths are computed.

For the capacity assignment process we distinguish between two basic types of traffic, elastic and stream traffic, which are associated with data applications and real-time services, respectively. Since their characteristics and QoS requirements are different, two distinct approaches are presented.

The fundamental property of elastic traffic is its rate adaptability, which is caused by the feedback mechanism of TCP. The transmission rate of a sender is adjusted to the available bandwidth along the route towards the receiver. As a consequence, the overall throughput, which is the relevant QoS measure for elastic traffic, depends not only on the link capacities, but also on the number of concurrent flows. In this dissertation, the theory of processor sharing is applied to the dimensioning of networks for elastic traffic. In our context, processor sharing is interpreted as ideal bandwidth sharing among flows, whose arrival instances form a Poisson process and whose service times are generally distributed. The basic model is investigated and deficiencies are identified. In networks with longer round trip times the

simple model underestimates the bandwidth needs. Therefore, we propose an extension, which incorporates relevant mechanisms of TCP and, thus, increases its accuracy. Finally, the dimensioning models are validated through extensive simulations using the network simulator *ns2*.

In case of stream traffic we focus on session-oriented services, which require call admission control (CAC) in order to guarantee the desired QoS. Capacities need to be assigned in a way that call blocking probabilities do not exceed a certain threshold and that packet-level QoS of accepted flows is guaranteed throughout their duration. Since call admission control architectures for IP networks are still a matter of research, we derive a set of abstract models that incorporate dimensioning-relevant properties. The decisive criterion for classification is the location of bandwidth components, which are considered for admission control (CA-controlled bandwidth components). Based on this criterion, three main categories of CAC schemes are distinguished and an appropriate two-step dimensioning strategy is presented for each of them. At first, only the CA-controlled bandwidth components are dimensioned. Based on the reduced load approximation and the Erlang fixed point equations, a non-linear optimization problem is formulated and solved. In the second step, the non-CA-controlled bandwidth shares are determined in a way that each of them is able to carry the worst-case traffic load. The corresponding optimization problem is solved through linear programming.

Overall, the algorithms and methodologies of this dissertation help network service providers to optimize their network infrastructure in order to reduce expenditures, while still being able to offer the desired quality of service.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Over the past ten years the Internet has experienced an unprecedented development. Within less than a decade it has evolved from a specialists' technology for data networking to a global communication infrastructure, which is being used by a large number of people. Its great success could initially be attributed to only a few data-centric applications, mainly e-mail and World Wide Web (WWW). However, the rapidly growing popularity of these applications started a self-energizing process. The Internet Protocol (IP) has found its way into basically all fields of networking. It is considered the technology of choice for the integration of diverse services such as voice, multimedia, and data. By providing a common platform it will enable the interconnection of all types of networks, thus, promoting the convergence of information technology, telecommunications, and media distribution [Ebe01].

Unfortunately, the downturn of the economy throughout the past three years has considerably slowed down the evolution of networking. IP is not spreading as quickly as anticipated – the process of convergence has been delayed. It turned out that many expectations specifically concerning profitability of networking were too optimistic, if not at all unrealistic. Especially network service providers were struck hard and they are still facing difficult times. Their numbers had increased quickly and competition had become fierce, causing profit margins to plummet. In a largely tight market it has not been possible to keep up revenue and to make profit. As a consequence, capital spending has been reduced drastically, badly affecting the supplier industry. Overall, the euphoria of the 90's has given way to a collective depression.

Nevertheless, it is unquestioned that the importance of IP will further increase and that it will serve as a platform for more and more services, requiring different types and degrees of Quality of Service (QoS). However, the awareness of cost-efficiency and profitability has risen. In order for businesses to succeed in the current tense situation, tight profit margins have to be compensated by efficient design and use of network infrastructure. Excessive overdimensioning for example, which by some people was seen as the general solution to any QoS-related problem, is not a viable option anymore. Even if the economic situation improves over the next years, network service providers as well as enterprises running their own networks cannot afford to waste resources. Networks will have to be designed appropriately for the types of services, which are to be offered (*network planning*). Furthermore, there is the need for network optimization methods, which allow providers to use the existing infrastructure as efficiently as possible in order to defer costly extensions (*traffic engineering*). *Capacity*

assignment and *routing optimization* are two central elements of network planning and traffic engineering – essential to master the challenges posed by emerging applications and services.

1.2 Important Drivers of Future IP Technology

In the following paragraphs, we present a selection of important applications, services, and initiatives, which are expected to greatly influence the development and the use of future IP technology. We identify important characteristics and point out their implications for network operation as well as network planning. The first two applications, *peer-to-peer networking* and *multimedia over IP*, represent current trends in today's Internet, which make use of the widespread availability of broadband technology and increased computer performance. The popularity of both fields of application has rapidly grown among Internet users. As it can be anticipated that this growth continues, peer-to-peer networking as well as multimedia communication will soon make up a large portion of the traffic transported in IP networks. The other two examples, *Next Generation Networking* and *Future Mobile Networking*, are initiatives of the telecommunications industry. They aim at the implementation of IP technology as the underlying transport infrastructure in order to facilitate the creation of new services and to realize cost savings. Originally circuit-switched telecommunication networks are partly turned into packet-switched network domains. Thus, IP technology will not only be applied in its native field of information technology but will also be utilized to support the convergence with the area of telecommunications.

1.2.1 Peer-to-Peer Networking

Throughout the 90's, the web was the main driver of the Internet, contributing the majority of transferred traffic [TMW97]. Currently, it still accounts for most of the traffic. However, over the past few years *peer-to-peer* (P2P) file-sharing applications have gained great popularity. On some links they have already passed WWW as the main traffic source [FML+03].

In a peer-to-peer environment, the participating hosts establish an overlay network typically without any geographical restrictions or even any understanding thereof. Each host arbitrarily connects to a number of other hosts and, then, takes on server as well as client functionalities (*servent*). In the case of file-sharing this means that P2P applications support the provision of files to the community by acting as server as well as the download of files by acting as client. When a user searches for a specific file, the query is broadcasted into the P2P network and servents, which have an appropriate match, reply. Query and reply messages are sent along the paths in the overlay network. In order to download a file, a direct connection is established between the two respective servents. Typically, the Hypertext Transfer Protocol (HTTP) [BLFF96, FGM+99] is used for this data transfer. All messages exchanged between the servents (signaling as well as data transactions) are carried over the Transmission Control Protocol (TCP).

The P2P concept can also be applied to applications other than file sharing such as instant messaging or distributed computing. It is even conceivable that in the future the P2P principle will be used for interactive, telecommunications-like services such as telephony or video conferencing. Instead of relying on directories or proxy servers, communication partners could find each other through peer-to-peer networks [Sch02].

1.2.2 Multimedia over IP

Although originally designed for data exchange, IP is increasingly being used for multimedia services. The reason for this development is the relatively cheap availability of high-performance IP technology within the networks (e.g., high-bandwidth core routers, increasing capacities in the access) as well as at the customer premises (e.g., low-priced high-performance computers).

The currently most popular multimedia applications in the Internet are audio and video streaming. According to [FML+03], streaming applications contribute about 1 to 6% of the overall traffic. In addition to streaming applications, two other types of multimedia services are emerging: voice over IP (VoIP) and video conferencing systems. However, in current IP networks these applications are not as widely used, yet. One reason is that their quality of service requirements are higher than what can be offered with today's employed technologies. While streaming applications can cope with varying bandwidths, longer network delays, and short outages, IP telephony and video conferencing systems need real-time delivery of data in order to function well. For example, the end-to-end delay for interactive communication should not exceed 150 ms [ITU99]. Furthermore, other important issues, such as security and charging are yet unsolved and, therefore, defer the commercial deployment of these services [LHJ+00]. Nevertheless, it can be expected that in the near future VoIP and video conferencing will play an important role in IP networks. The foundations for multimedia networking have been laid. To achieve compatibility between different types of end systems, two major multimedia frameworks have been specified. The International Telecommunication Union - Sector Telecommunication (ITU-T) has created H.323, which provides a framework for real-time services in an IP environment [ITU00b, KKS01]. On the other side, the Internet Engineering Task Force (IETF) promotes the Session Initiation Protocol (SIP) as a versatile and extendible signaling protocol for service session control [RSC+02].

1.2.3 Next Generation Networking

Next generation networking (NGN) refers to the latest development in telecommunication networks in order to realize open network standards in a deregulated market. The idea behind NGN is a service-centric approach to heterogeneous networks [Coc02]. The provision of services is decoupled from the transport infrastructure by introducing a distributed processing environment (DPE). This middleware layer provides a homogeneous view of underlying network technologies and facilitates the development and implementation of new cross-platform applications [SKK00a]. To enable third-party application providers to access telecommunications services of different network operators, application programming interfaces (API) such as Parlay [ETS03] and JAIN [JTG00, MBD00] are being standardized. Thus, the separation of the overall network architecture into a service and a transport domain with the middleware layer as the "glue" allows greater flexibility of service creation in an open telecommunications market.

First realizations of NGNs focus mainly on telephony services. They implement the functionalities of traditional telephone switches in a distributed way, with IP as the underlying transport infrastructure (Figure 1.1) [MM00]. In order to provide interoperability with conventional public switched telephone networks (PSTN), media gateways are located at the edge of the IP transport domain, which carry out media mapping between circuit-switched and packet-switched technologies. New calls and sessions are signaled to a central unit, the

media gateway controller, which performs session control functions such as call admission control. For the communication between media gateways and media gateway controllers the Megaco/H.248 protocol has been specified [GRR00, CGR+00].

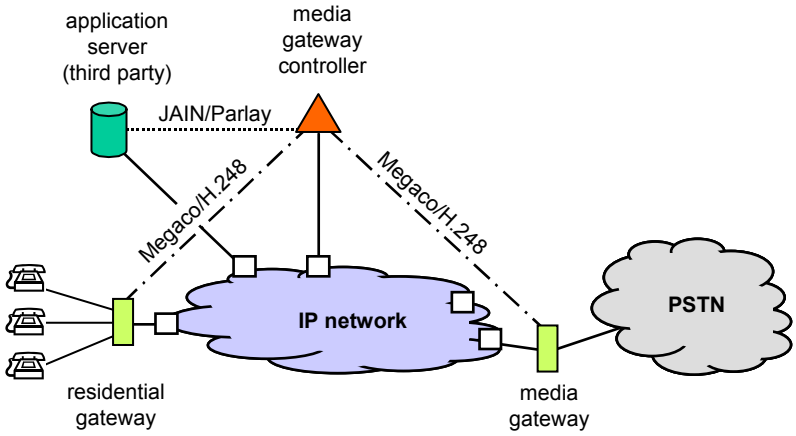


Figure 1.1: Schematic NGN telephone architecture

1.2.4 Future Mobile Networking

The convergence of IT and telecommunication services is not restricted to fixed networking. Data services have also become an integral component of wireless telephone systems [VLLX02]. To enable efficient data transmission, once purely circuit-switched cellular systems such as GSM (*Global System for Mobile Communication*) have been enhanced by packet switched bearer services (e.g., *General Packet Radio Service*, GPRS) [EVB01]. Coming generations of mobile networks (3G and beyond) will realize seamless provisioning of voice and data services. For these future wireless networks, IP is considered a key technology since its adoption as the network protocol is expected to create substantial synergies [GDF00, LPHC02, KJC+03]. Wireless networks with IP at the air interface will allow network service providers to use a common backbone for different types of access technologies. Furthermore, an all-IP environment facilitates the creation of new applications and services. The ultimate goal is that a converged mobile network architecture makes use of various wireless access technologies including cellular networks, wireless LAN's as well as satellite connections [ZVTZ02].

1.3 Implications for Network Operation and Planning

The new applications and services will change the nature of traffic in future IP networks, having an impact on the amount of traffic and its distribution as well as on the flow and packet-level characteristics. Furthermore, the rising demand for real-time applications will create the need for QoS-enabling technologies. The consequences for network planning and optimization are pointed out in the following paragraphs.

Traffic quantity

Over the past few years, P2P file-sharing systems as well as audio and video streaming applications have already lead to significant traffic growth in Internet Service Provider (ISP) networks. With ongoing deployment of high-speed access technologies in the last mile, this trend is expected to continue (assuming that the legal situation concerning P2P file-sharing applications does not change). Especially, P2P networks have contributed a great portion to the traffic increase. In [SW02] for example it was reported that the amount of P2P traffic in a large ISP backbone network (provider undisclosed) more than doubled in just a two-months period (November/December 2001). The majority of users were connected with bandwidths higher than modem speed (56 kbps). Measurements at a commercial streaming site [vdMSK02] reveal a similar trend for audio and video streaming. Most of the observed streaming sessions were established selecting a higher bandwidth version, using 250 kbps instead of 56 kbps. Overall, the high bit rate streams accounted for 95% of the traffic.

The "bandwidth greedy" applications pose great problems for network service providers. In case of web traffic, it is possible to reduce traffic load within the network by setting up proxy servers at the edge. Due to the distributed nature of P2P applications, the traffic growth problem cannot be alleviated by installing cache servers. Therefore, in order to avoid traffic congestion within the network, network capacities need to be expanded. If the traffic increase is confined to certain parts of the network, it might also be sufficient to optimize the routing of the traffic flows (see next paragraph). However, the improved performance could in turn lead to higher user activity levels and even more traffic, which in the end fully compensates for the increased bandwidth. As long as service providers do not have a possibility to differentiate between individual traffic types and to set service-specific limits, there is no other solution than increasing bandwidth regularly.

Traffic distribution

Intuitively, it is expected that P2P applications as well as mobile services make it difficult to predict the traffic distribution in the network, due to the dynamic nature of both application types. Hosts join and leave the system continuously.

However, for P2P systems it has been observed that not all participants contribute equally to the overall generated traffic. Over 90% of the total traffic is transmitted by only 10% of the hosts, and the top 1-2% of the hosts still account for over 50% of the traffic. The same skewed behavior can be observed for the receiver side, where the top 0.1% consumers receive 25% of the total traffic [SW02]. Even more surprising is that despite the dynamic nature of P2P applications at micro-level, the observed traffic pattern at prefix level has been quite stable. Thus, the overall system is very similar to client-server environments.

For coming generations of mobile networks it is not yet possible to predict typical traffic patterns, as these will depend on the popularity of future services and the prevalent application mix (telecommunications vs. data). Nevertheless, it can be expected that telecommunications services such as telephony will lead to similar usage patterns as observed in today's cellular networks. This would mean that the call traffic characteristics per user (e.g., mean call rate and call holding time) are geographically homogeneous, while the amount of traffic including mobility-related signaling traffic depends on the type of cell (e.g., metropolitan cells or cells containing high density routes) [BF97]. For mobile data services similar considerations could be done, suggesting flow patterns typical for client/server networking.

Concerning the mobility behavior of users in mobile networks, measurements in local-area as well as metropolitan-area wireless data networks have shown that most users are stationary [TB00, TB02].

The fact that traffic distribution cannot be easily predicted before the network is in operation stresses the need for network optimization methods that are applied as network management tasks (*traffic engineering*). Once the traffic distribution is determined by means of monitoring, routing optimization can be used to adapt the traffic flows in the network in a way to better utilize the available network capacity. The observation that P2P as well as mobile networking induce rather stable traffic patterns encourages routing optimization as a means of traffic engineering.

Traffic flow characteristics

The presented applications have different effects on the traffic flow characteristics. The majority of traffic in P2P networks, which is generated by file-sharing transactions rather than signaling, is in principle very similar to web traffic: small requests sent into one direction are followed by large data transfers into the opposite direction. The results are highly asymmetric traffic flows concerning the transferred amount of data. For web traffic, upstream/downstream ratios of 1:8 in average and up to 1:30 have been observed [Cha00]. Since longer downloads lead to higher degrees of imbalances, the asymmetry of flows caused by P2P file-sharing applications is more pronounced. While the average WWW page (including associated objects) is about 20 to 60 kbytes [RLGPC$^+$99, Cha02], the average size of transferred files in P2P systems is currently around 3.7 Mbytes [SGG02]. Streaming applications also generate highly asymmetric traffic streams, with the majority of the data flowing from the server to the client [MH00]. The amount of traffic depends on the encoding scheme, the preferred bit rate, which can be set by the user, as well as the duration of the streaming session. Finally, communication-oriented, interactive services such as telephony and video conferencing, typically establish bi-directional sessions between two or more hosts. As a consequence, the resulting traffic flows between the end systems are symmetric, unless, the users choose different coder settings and bit rates [GCR00].

The asymmetric property of IP traffic needs to be considered during network planning and optimization processes. Especially for the dimensioning of networks this characteristic can be exploited for the sake of cost savings. By taking into account unidirectional links, the required capacities can be assigned more accurately. This distinction regarding direction is expected to have a greater impact in networks with lower traffic aggregates (such as in access networks) where the characteristics of individual flows have stronger influence on the overall traffic aggregate. In backbone networks the per-flow traffic asymmetry is less pronounced due to the superposition of a great number of flows.

Quality of Service requirements

Traditionally, IP networks have only been able to offer *best effort* service: all traffic flows share the same resources (bandwidth, buffer space) and as a consequence receive similar quality of service (depending on the applied transport protocol). This was never a problem as long as the great majority of applications were data-centric and, therefore, had similar requirements. However, with the emergence of diverse services this situation has changed. If IP is to be

used as a unified network platform, it will be necessary that adequate service-specific QoS can be provided. Therefore, following fundamental functionalities will have to be realized in future IP networks:

- *Differentiated packet treatment*: In order to offer several classes of QoS, network devices have to be able to handle IP packets in a differentiated way, corresponding to the class of service they belong to.

- *Resource reservation and admission control*: Whenever network resources are limited and strict QoS requirements have to be met, the usage of resources needs to be controlled, monitored, and regulated.

- *End-to-end service provisioning*: QoS has to be provided on an end-to-end basis, making it necessary to coordinate service provisioning across multiple domains and provider networks.

Over the past years, numerous mechanisms have been developed, which provide important building blocks for the overall QoS framework [GP99, LF03]. Furthermore, various concepts have been proposed, which incorporate fundamental QoS mechanisms within one architecture in order to realize comprehensive QoS-enabled networks. Examples are IETF's *Integrated Services* [BCS94] and *Differentiated Services* [BBC+98] frameworks, as well as resource manager and bandwidth broker concepts such as [GM02], [EGK+03], or [TAP+01].

It is obvious that QoS aspects, i.e., the various notions of QoS as well as the enabling technologies, have to be considered for network planning. They determine the constraints and the objective of the relevant optimization problems. As soon as QoS requirements have to be met, the networks need to be dimensioned appropriately. However, it is not possible to take into account every technological detail as this would make the planning process too complex. Therefore, appropriate abstractions of the relevant technologies and QoS requirements are essential.

1.4 Contributions of this Dissertation

In this dissertation network planning is investigated for future IP networks with multiple services. Specific focus is put on routing optimization and capacity assignment as these are key processes, which are needed to guarantee the optimal functioning of an IP network. As we will see in the second chapter, these subtasks arise at different stages of network planning and operation processes and at various times throughout a network's life cycle.

Routing Optimization

Routing optimization plays a crucial role for network planning (initial design and extension planning) as well as for network operation (traffic engineering). Given a certain traffic demand matrix, it is the objective to optimize QoS only by modifying the routes of traffic flows through the network and not by changing or extending the network infrastructure. Two different approaches are presented in this dissertation.

- In the first case, we rely on **conventional, metric-based routing protocols** such as OSPF and EIGRP (see Chapter 3 for a detailed description). With these protocols the routes of traffic flows can be influenced by setting the link metric values appropriately. The resulting optimization problem (*weight setting problem*) is formulated as a **mixed-integer program**, which can be solved for small networks. For larger topologies, a heuristic algorithm is presented, which is based on the **genetic algorithm** framework. A significant novelty of our routing optimization approach is the **combined consideration of additive and concave link metric types**.

- The second form of routing optimization makes **use of source-routing**, which in IP networks is mostly realized through MPLS (again, see Chapter 3). However, unlike other proposals for MPLS routing optimization, we consider MPLS a complementary technology and use it **in combination with conventional routing protocols**. In our approach, most of the traffic is still subject to native IP routing, where the corresponding IP packets follow metric-based shortest paths. Only a **small set of traffic flows is selected and routed along explicit paths**. This way, the administration effort can be kept lower since it is not necessary to establish a dedicated path between each pair of network nodes. Furthermore, the costs for new equipment can be reduced since not all routers need to be capable of source-routing.

Capacity Assignment

The emergence of diverse services each having specific QoS requirements makes it necessary to divide the link capacities into bandwidth shares and to assign these to the respective traffic classes. Similar to routing optimization, capacity assignment needs to be carried out at different stages of network planning and operation. It has to be done when the network is initially set up, but also during network operation (e.g., when the traffic situation changes due to increased popularity of some applications or when new services are added to a provider's portfolio). In this dissertation we differentiate between two fundamental types of traffic, elastic and stream traffic, and present capacity assignment strategies for each of them.

- **Elastic traffic** is generated by non-real time applications and is carried by the Transmission Control Protocol (TCP), the main transport protocol for data in IP networks (see Chapter 4 for a detailed discussion). We present a dimensioning model, which is based on the processor sharing model. It can be applied for capacity assignment in IP networks, taking into account the desired level of QoS. For elastic traffic, the **relevant QoS measure** is **throughput**, which is achieved for data transactions. The main contribution of our work is the **extension of the processor sharing model to incorporate important TCP features**, and to **demonstrate the applicability** of the theoretical model through **extensive simulations**.

- **Stream traffic** is associated with real-time services, which have strict QoS requirements concerning the transport of packets over IP networks. For these applications the packets have to be delivered in a timely manner. In this dissertation it is assumed that this type of traffic **is subject to call admission control** (CAC) in order to be able to guarantee the desired QoS. Therefore, we **investigate various CAC schemes** and derive **abstract models**, which provide the basis for the capacity assignment process.

The **corresponding optimization problem is solved in a two-step procedure**, which incorporates non-linear and linear optimization.

1.5 Outline

In Chapter 2 we provide an **overview of network planning and optimization processes** for IP networks, considering multiple services. We structure the processes and show how the overall problem can be mapped onto individual optimization approaches.

In Chapter 3 the new strategies for **routing optimization** are presented, deploying conventional routing protocols (OSPF and EIGRP) as well as source-routing protocols (MPLS). Specifically, we introduce a traffic engineering approach, which utilizes the potential of multiple metric types as supported by EIGRP. The problem is formulated as a linear programming model, and a heuristic algorithm based on the genetic algorithm framework is proposed. Furthermore, we investigate how routing optimization based on conventional routing protocols can be improved by partial source routing as enabled by MPLS. The optimization problem of this hybrid routing approach is also solved through linear programming.

In Chapter 4 the **dimensioning process for elastic traffic** is investigated in detail and a new model is presented. Based on the idea of processor sharing, we propose a dimensioning model, which considers TCP-specific mechanisms. Through simulations it is shown that this model is able to capture relevant characteristics. Furthermore, we demonstrate that the processor sharing model can be used for network dimensioning.

Chapter 5 focuses on **dimensioning for stream traffic** with hard QoS requirements. Different call admission control schemes are investigated and their implications for network dimensioning are discussed. In order to derive an optimized dimensioning strategy for each of them, the individual schemes are abstracted and the optimization problem is specified. Since the overall problem is too complex to be solved, we propose a two-step methodology, consisting of a non-linear optimization step and a linear programming approach.

Chapter 6 gives a **summary** of this dissertation and discusses the most important results. Additionally, interesting issues for further research are pointed out.

Chapter 2

Overview of IP Network Planning

In this chapter we provide an overview of optimization issues, which arise in planning and operation of IP networks with multiple services. At first, the scope of network planning is defined and relevant optimization objectives are presented. We then identify and describe important network planning and management tasks and show how routing optimization and capacity assignment fit into the overall picture. The implications of multiple services are discussed, and their consideration in our planning and optimization procedures is pointed out.

2.1 Scope of Network Planning

The global Internet is organized in individual building blocks, the so-called *autonomous systems* (AS). An AS is loosely defined as "a connected group of one or more IP prefixes run by one or more network operators, which has a single and clearly defined routing policy" [HB96]. In January 2002, for example, about 12500 ASes and over 100000 network prefixes existed. The largest ASes comprised over 100 prefixes, which represent individual networks or subnets [BNC02].

The scope of network planning is typically limited to one such network, which either makes up an AS by itself or is just part of a larger AS. Although it is conceivable that networks belonging to different providers are planned or optimized in a coordinated fashion, thus, achieving more cost-efficient or higher-performance infrastructures, this type of cooperation is rather unrealistic. Networks are usually planned and operated independently. Where needed, contracts between providers regulate peering relations or service agreements.

Figure 2.1 shows the hierarchical network structure of the Internet in the early 90's [CB92, HWB93]. Although the Internet has grown drastically since then, a similar structure still exists today. The backbone of the Internet is made up by transit networks (*Tier-1 IP backbones*), which are specifically designed for carrying data over far distances. These networks can be of commercial nature such as the global backbones of AT&T or Sprint, or research driven such as GÉAN, the European research network operated by DANTE. Connected to these backbone networks, regional Internet service provider networks (*Tier-2 ISP*) provide connectivity to research institutes, universities, enterprises, and private customers, which make up the lowest level of this global hierarchy. In the following paragraphs, typical network structures and realizations are described.

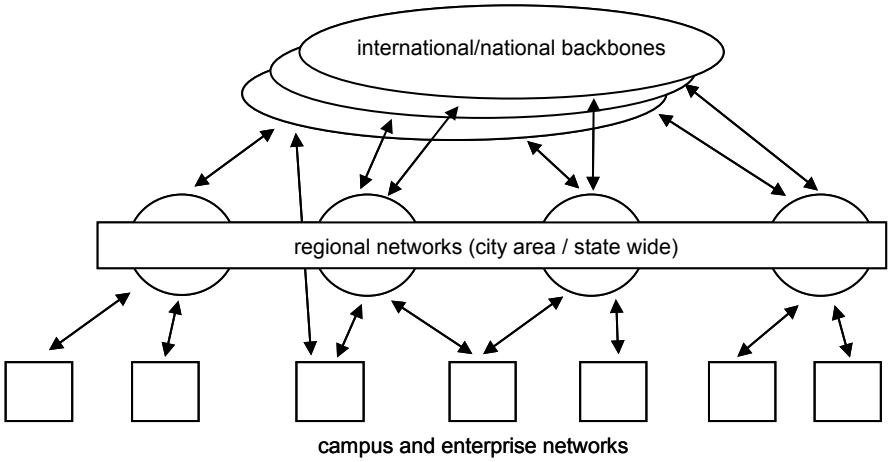


Figure 2.1: Schematic Internet architecture

Enterprise and Campus Networks Enterprise and campus networks typically have star or tree-shaped topologies. The backbone consists of only a few routers or switches (often just one), which might be fully meshed. From there, links go to the different buildings or departments on site, where additional switches connect users either via floor switches or directly. The access to the public Internet is provided through regional or national ISPs. Depending on the size of the enterprise, various technologies such as dial-up connections, digital subscriber lines (DSL), or leased lines can be employed.

If an enterprise has several sites, they are usually interconnected through leased lines or by means of a virtual private network (VPN). In the first case, the company leases circuits or dedicated communication services from public network operators, which are then at its exclusive disposal. Due to its cost efficiency, a hub-and-spoke structure is often chosen where one central site provides the interconnection of all others. In case of a VPN solution, the interconnection of different locations is bought as a service from an ISP. The structure of the actual VPN topology is not relevant to the enterprise, as it relies on the ISP to provide the appropriate service, which is usually specified in a service level agreement (SLA).

Regional Networks Figure 2.2 shows a generic model of a larger ISP, which provides Internet access and services to private and corporate customers, but also has its own backbone network (thus, functions as network service provider). Customers are connected through local offices, which host termination routers for dial-up connections, DSL, or leased lines [Hus99]. From there, traffic is routed to distribution nodes situated in regional offices. Usually star or ring networks are used for this connection. In case of a star structure, redundancy might be achieved by using backup lines to either the same regional office or to another one. In regional offices the traffic coming from different local offices is aggregated and routed to backbone routers, being located in regional or central offices. Often, there exist connections to at least two backbone routers. The backbone itself is realized in form of a meshed topology. At *peering points* traffic is exchanged with upstream or neighboring ISPs.

The functionality of the backbone network can be divided into a *core* and a *distribution* layer. The core technology is specialized in fast switching, while administrational tasks or

routing adaptation issues are left to the distribution layer.

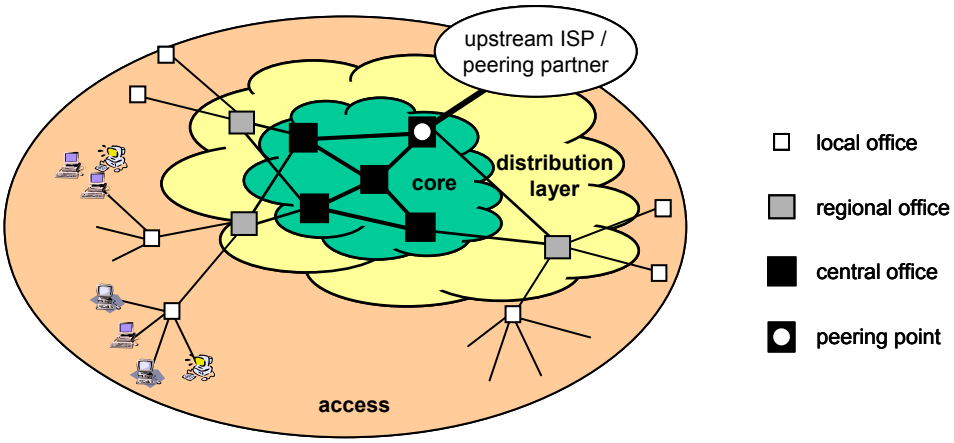


Figure 2.2: Typical ISP network structure

International Backbones International backbones are in principle networks of large or specialized ISPs. Instead of providing access to private customers or corporations, they mostly carry transit traffic coming from regional and national ISPs. For example, GÉANT connects national research and education networks such as the German G-WIN, which in turn provide access to universities. However, the distinction between regional and international networks does not mean that the two networks cannot belong to the same provider. Many of the large ISPs offer access as well as transit services. Still, the different purpose and functionalities of these networks as well as the employed technologies allow a differentiation of Tier-1 and Tier-2 networks.

The algorithms and procedures presented in this dissertation apply to various parts of the overall Internet hierarchy:

- Routing optimization (Chapter 3) focuses on networks, which are typically associated with the upper two hierarchy levels, i.e., which provide transit service to customers or service providers. The meshed network structure of backbones is a prerequisite for routing optimization as this requires several path alternatives between two nodes. In tree-shaped enterprise networks with only one path option between any two hosts routing optimization is not applicable.

- The dimensioning procedure for elastic traffic (Chapter 4) is most relevant for lower-capacitated links and networks, such as access networks, connections between local and regional offices, or leased-line enterprise backbones. These are the areas where capacity is most expensive. Therefore, a thorough capacity assignment plan is crucial as it allows great cost savings.

- The proposed dimensioning methodology for stream traffic (Chapter 5) applies to all types of networks. This could be general ISPs and enterprises that offer real-time services with hard QoS requirements to customers or employees. It could also be specialized network service providers or mobile network operators who are running NGN-style

IP backbones. The main criterion for the applicability of our methodology is that a QoS-enabled network with CAC mechanism is employed.

2.2 Objectives of Network Planning

In order to perform any kind of optimization, a clear objective is necessary, which allows the quantitative evaluation of a solution. As the ultimate goal of every service provider is to make profit, it is necessary to have measures, which reflect the monetary inflows and outflows associated with network operation. For both positions we take a rather simplistic view as our focus is on optimization and not business-related issues. In this dissertation it is assumed that the income is correlated with the quality of service that a provider is able to offer: the better the service, the more customers will use it and, therefore, the more revenue is generated. The outflow is represented by costs and expenditures necessary to build and run a network. Thus, *cost* and *quality of service* can serve as objectives for optimization processes within the context of network planning and operation: costs have to be minimized while QoS is maximized. However, since the two measures are usually negatively correlated, a service provider has to find the right balance in order to succeed in tight markets.

In the following paragraphs we introduce various types of cost and QoS measures and discuss how they are of concern for our optimization processes.

2.2.1 Network Costs

The evaluation of network costs is a complex matter. However, to keep optimization problems manageable, we assume rather simple abstractions of cost terms. Usually two categories of expenditures are differentiated, which can loosely be associated with building a network (capital expenditure, *CAPEX*) and with running it (operational expenditure, *OPEX*). CAPEX includes items such as equipment, capacity, installation, and replacement, while OPEX is related to expenses such as rights of way, rentals, and operations and maintenance. Table 2.1 lists some examples for the two cost categories.

CAPEX	**OPEX**
equipment	rights of way
cable	building rental
civil works	operations and maintenance
installation	management and salaries
replacements	cost of inventory
customer premises equipment	marketing
licenses, permits	overheads

Table 2.1: Sample cost components

In this dissertation, the interpretation of CAPEX is reduced to the cost for link capacities, bandwidth allocation, and routing devices. OPEX, which in general is quite difficult to quantify, is only considered in a qualitative manner. Our reasoning is that OPEX can be reduced, as long as network configuration is kept simple. Therefore, we concentrate on network

optimization issues, which can be based on existing and conventional technologies and which reduce the administrational effort.

2.2.2 Quality of Service

Throughout this thesis the term *quality of service* is used as understood by the IETF. It refers to service requirements that need to be met by the network while transporting a traffic flow [CNRS98]. This interpretation is not the same as the one given by the International Telecommunications Union (ITU) and the European Telecommunications Standards Institute (ETSI). For ITU/ETSI, *QoS* is the quality perceived by the user [ITU93].

As stated above, QoS refers to packet transfers or flow transactions. However, if QoS is used as the objective of a network optimization problem, a measure is needed, which quantitatively reflects the overall QoS achieved by a certain network configuration. It has to provide one value, which expresses the quality of a network respective to QoS. For further reference, this is called *network QoS*. In the following paragraphs, various measures for flow-based QoS as well as for network QoS are presented.

2.2.2.1 Flow-based QoS Measures

Typical flow-based QoS measures are packet delay, delay jitter, packet loss rate, and throughput.

Packet Delay The end-to-end delay is the total time of an individual packet transfer from the source to the destination. It contains delay portions that arise from processing, serialization, propagation, and queuing. While per-packet delay is an important QoS aspect for real-time traffic, data traffic is not much affected by it as long as the overall transaction speed is high enough (see *throughput*).

Delay Jitter Delay jitter refers to the variance of individual packet delay values. It is the difference between the largest and the smallest end-to-end delay of a sequence of packets. This measure is mainly important for real-time traffic such as voice where a certain sample rate should be kept up from the source to the destination. Streaming applications deploy play-out buffers in the receiver, which are able to equalize irregularities of interarrival times.

Packet Loss Rate Packet loss rate is an important QoS measure, both for data applications as well as for real-time services. Since data transmission requires the correct receipt of all packets, packet losses cause retransmissions. These slow down the overall transfer process and decrease the perceived QoS. For real-time traffic packet retransmission is often not possible. Therefore, packet losses might lead to unacceptable service degradation.

Throughput This measure is mostly used for data traffic where it is not important how fast the individual packets reach their destination but how long it takes to transfer a certain amount of data. Only after the last byte arrives at the receiver, the data transaction is completed. In order to emphasize the difference between gross and net data rates (with and without payload overhead and retransmissions, respectively), *goodput* is often used to indicate net throughput.

2.2.2.2 Network-relevant QoS Measures (*Network QoS*)

The flow-based QoS measures can also be used to quantify the quality of a network. However, since the number of flows in a network is usually large, it is necessary to summarize the individual QoS values (one per flow) in an appropriate way and form one value with network-wide relevance. Possible network QoS measures, which are derived from flow-based QoS, are:

- average values, e.g., average packet loss rate over all flows in the network
- extreme values, e.g., maximum packet delay of all flows in the network

Often, it is too complex or not possible at all to determine flow-based QoS parameters in order to derive averages or maxima. In these cases, network QoS measures are required, which are based on traffic aggregates rather than individual flows. One such parameter, which relates traffic aggregates to QoS, is simply the average link utilization. It directly affects packet loss rates, queuing delay, and achievable throughput and, thus, can serve as an appropriate, link-specific QoS measure. In order to derive one network-wide QoS value, many different functions are meaningful. The two most common utilization-based network QoS measures are:

- the maximum link utilization in the network
- the sum over all links of *f(utilization)*, where f is a convex function over the link utilization (e.g., waiting time formula for M/M/1 queuing model).

In both cases, the quality of a network solution improves with decreasing value of the respective QoS measure.

In this dissertation different QoS measures are applied as objectives or constraints. For routing optimization, the maximum link utilization is used as the optimization objective. For dimensioning, average flow-based measures are considered as constraints: for elastic traffic this is the average throughput, and for stream traffic the average blocking probabilities are relevant.

2.3 Optimization Issues

In this section, an overview of network planning and optimization is given, considering multi-service specifics. The presented tasks apply generally to any type of IP network, although the focus and the importance of the individual steps varies from case to case.

Figure 2.3 illustrates optimization issues within the context of network design and network management, revolving around network operation. With *network operation* we subsume all processes and actions necessary to run a network and to keep it in a stationary functional state. While this, of course, includes a great number of tasks such as user administration, accounting, security engineering and many more, we focus on issues, which guarantee a faultless and satisfactory service concerning the network as a transport platform. Assuming, a network is configured and is running and customers are sending data through the network, actions that assure a desired QoS are *traffic control*, possibly *call admission control* (CAC), and – in case of temporary failures – *rerouting*. Traffic control refers to packet-level measures that need to

be taken in order to achieve service-specific appropriate QoS for all traffic flows traversing the network. It includes technologies such as packet marking, scheduling, buffer management or traffic policing. Since active traffic flows share the limited network resources, call admission control might be necessary to keep the network from being overloaded and services from experiencing quality degradation. Finally, a fundamental issue is to provide connectivity by means of a consistent routing scheme. When topology changes occur due to link or node failures, routing needs to be adapted.

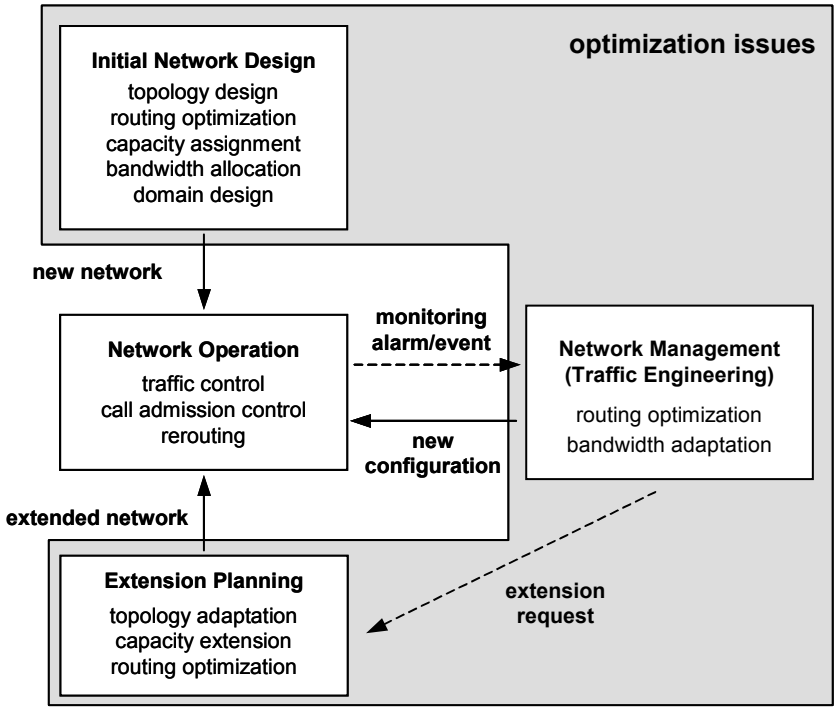


Figure 2.3: Planning and optimization issues revolving around network operation

Taking network operation as the center of our attention – as this is actually the process, which generates revenue for network service providers – numerous optimization processes are necessary to get a network started and to keep it up over a period of time. First of all, it is necessary to initially set up a network. *Initial network design* incorporates several optimization issues, which will be discussed in the following sections. After having built the network, its operational state has to be continuously monitored in order to check and verify whether it conforms to the requirements. As soon as a non-satisfactory operation state is indicated, countermeasures have to be taken. In our case, such non-satisfactory states relate to traffic overload situations, which might arise due to increasing traffic quantity or temporary traffic variations. If possible, the overload situations can be alleviated through *network management (traffic engineering)* actions, such as routing optimization or adaptation of bandwidth assignment. In either case the new configuration is uploaded to the network devices and network operation is continued. However, if the existing infrastructure is not sufficiently dimensioned to carry all traffic with appropriate QoS, the network has to be extended. This process is denoted as *extension planning*.

In the following sections we describe the individual optimization processes in more detail.

We show that they contain some basic modules, which are fundamental to various network-related design and management tasks.

2.3.1 Initial Network Design

As the network design process is too complex to be solved within one integrated optimization procedure, it is commonly split up into subtasks, which are then solved individually. Figure 2.4 illustrates the various tasks. Although the steps are listed in a sequential manner, network design is mostly not a straightforward process. Some of the steps are carried out in order to provide a basis for subsequent ones. Then, after solving the following steps, earlier ones are revisited. This iterative approach is indicated in the figure with loop-back arrows. Overview of network design issues, from a theoretical as well as practical perspective, can be found in several books such as [Ker93], [Cah98], [Rob98], or [McC98].

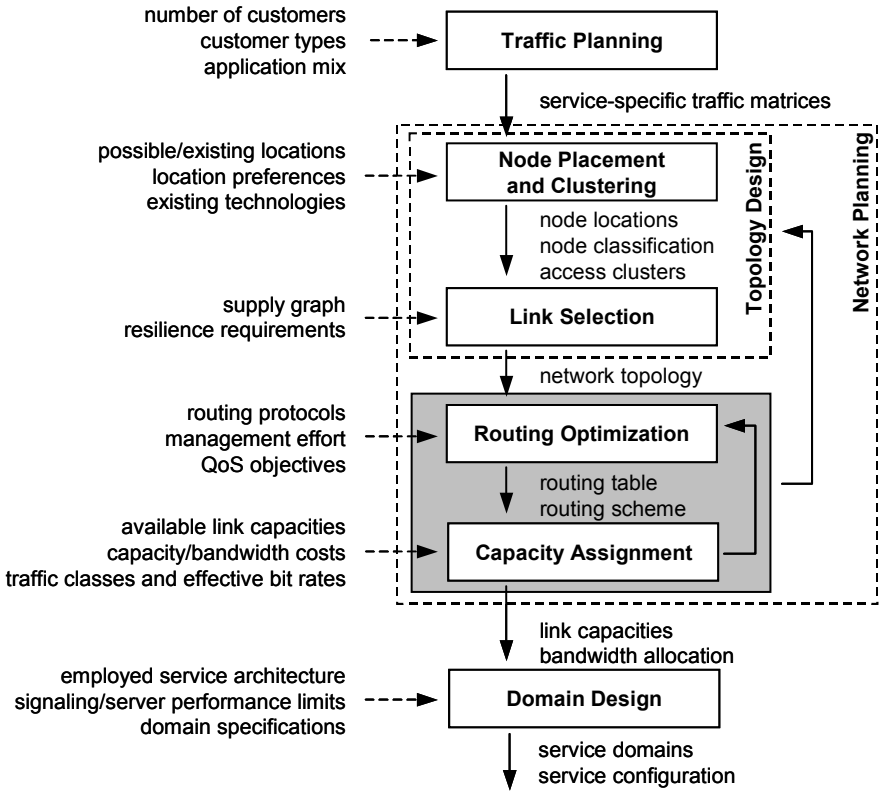


Figure 2.4: Initial network design process

2.3.1.1 Traffic Planning

At the beginning of the design process it is necessary to come up with an estimate for the traffic that will be carried by the network. In the context of network design, traffic planning comprises basically three tasks: *service classification*, *estimation of traffic quantity*, and *estimation of traffic relations* (demand matrix).

Service Classification The applications and services, which will be offered in the network, need to be classified respective to their required QoS. It is recommended that the most important applications and services are identified and assigned to a set of only a few categories, which are then mapped onto traffic classes. These traffic classes specify the required QoS that the network has to deliver for each of them. The implementation of the classes in the network, i.e., their association with appropriate network services, is technology dependent. Figure 2.5 illustrates this mapping process. In the given example, voice and gaming applications constitute a single traffic class since both demand low delay and low loss values. On network layer the Differentiated Services framework is employed, which provides the two special services *expedited forwarding* and *assured forwarding* [DCB+02, HBWW99] in addition to IP's standard best effort service. Figure 2.5 indicates one more classification scheme, which will be relevant for capacity assignment (see below): the distinction between stream traffic and elastic traffic.

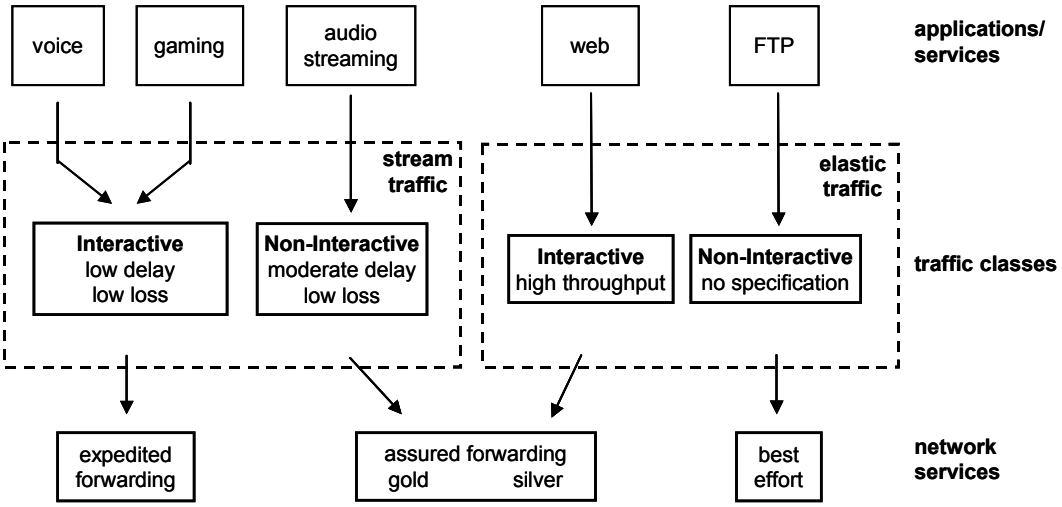


Figure 2.5: Application and service mapping process

Estimation of Traffic Quantity For each of the identified traffic classes, the geographic distribution of traffic quantity has to be estimated. This can be done by taking into account various aspects, such as economic and demographic data, technological trends, user behavior, or relevant growth rates [KO02, SKO+02].

Estimation of Traffic Relations Finally, service-specific demand matrices have to be derived. The rows and columns of a traffic matrix correspond to the edge nodes of the network, i.e., the points where traffic flows enter or leave the network. These points comprise Points of Presence (PoP), where customers access the IP network, network-internal server and proxy locations, as well as exchange points, through which connections to peering partners are established. However, since the edge points are usually not fixed, yet (see next paragraph), the level of detail of the initial traffic matrices is rather coarse. A row/column might summarize all expected residential users, corporations, internal servers, or neighboring networks. As the network design process progresses (i.e., customers get assigned to individual PoPs, locations of

servers, proxies and exchange points are specified), the traffic matrix can be steadily refined.

Due to the many elements of uncertainty, traffic planning at this point can only give rough estimates. Even if customer and traffic predictions were accurate, a new and popular application can quickly and drastically change the traffic quantity and the flow patterns and, thus, change the nature of traffic in the network (see for example peer-to-peer networking in Chapter 1).

2.3.1.2 Topology Design

Topology design embraces the tasks of *node placement*, *node clustering*, and *link selection*.

Node Placement and Clustering Given the potential customer distribution over the service area as well as the traffic demand, appropriate locations for access routers and Points of Presence have to be selected. Furthermore, distribution, core, and exchange nodes have to be placed. The choice of node locations affects traffic patterns in the network and, as a consequence, link capacities, link costs, and total network costs.

Looking at this task from an optimization point of view, the problem is typically solved by a clustering algorithm where customers are assigned to access routers and access routers to distribution nodes in a way that link costs are minimal. The link costs at this point are often assumed to be proportional to the link lengths. However, in practice there are usually limited possibilities for the locations of local, regional and central offices, as well as for the placement of access and core routers, exchange points, and servers. They depend on existing locations or the availability of appropriate buildings.

Link Selection After the node locations and the access clusters are determined, the individual nodes have to be interconnected. Within an access cluster typically a ring or star topology is chosen, while for the backbone network a mesh structure is preferred. A crucial factor for the link selection process is resilience. The decision whether or not redundant routes should exist and to which degree the network should be resilient (single/double link failures) greatly affects the network structure.

The problem of the link selection task is to find a cost-optimized network structure taking into account topological constraints. Usually, a set of candidate links is provided (*supply graph*), from which the network links have to be chosen. In order to derive the network cost at this point and, thus, evaluate the quality of a network structure, simple routing and dimensioning principles are assumed. Taking for example hop-based shortest-path routing and linear, continuous capacity costs, a certain network structure can quickly be evaluated by routing all traffic demands along the shortest paths and summing up the resulting link loads. Later on, after the steps of routing optimization and network dimensioning have been carried out, one can come back to topology design and try out different network structures.

2.3.1.3 Routing Optimization

Routing optimization describes the process of improving a network solution by finding advantageous path patterns while the network structure is not altered. In the context of network design, the quality of a routing solution is assessed after carrying out the subsequent capacity assignment process and determining the total network cost. Thus, routing optimization

and capacity assignment are interrelated, since the quality of routing cannot be evaluated without the results of the dimensioning process. As routing optimization needs to consider the constraints imposed by the employed routing protocols, a good understanding of routing technologies and their fundamental principles is essential. Relevant details will be presented in Chapter 3. Specifics of multi-service routing are discussed below in section 2.3.2.1.

2.3.1.4 Capacity Assignment

Based on the demand matrices for the individual traffic classes, the given network topology, and a fixed routing pattern, the capacities of the links have to be determined. It is the objective to minimize the overall costs while meeting the QoS requirements.

At this point, we assume that technologies are employed, which allow service-specific partitioning of link capacity. Through scheduling mechanisms such as Weighted Fair Queuing (WFQ) [Zha95] the total capacity of a link is divided up into bandwidth shares, which then are assigned to the individual traffic classes. Traffic policing units at the network edges (e.g., token buckets) assure that individual traffic flows within the traffic classes do not negatively interfere with each other. Furthermore, call admission control is applied to guarantee strict QoS to real-time critical services.

This assumption allows us to consider the different traffic classes separately and to determine their necessary capacity shares independently. In order to obtain the total capacity of each link, the individual bandwidth shares need to be summed up. Figure 2.6 illustrates the resulting link model [BRF02, RBF02].

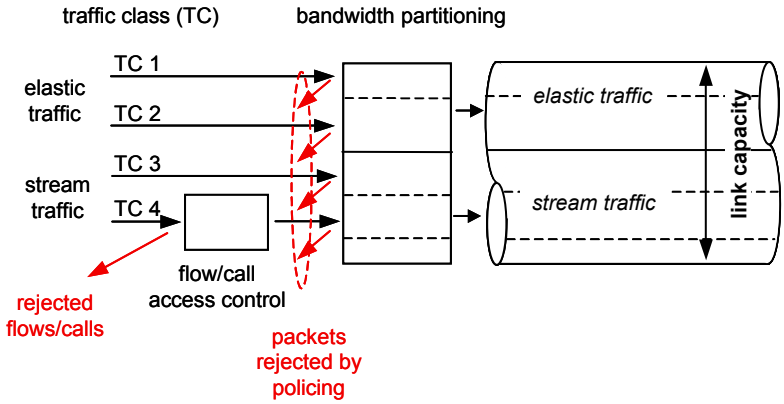


Figure 2.6: Link dimensioning model for multiple services

In the figure, the individual traffic classes are labeled as *elastic traffic* (data traffic) or *stream traffic* (real-time traffic), pointing out that these two fundamental traffic types require quite different dimensioning methodologies (see Chapters 4 and 5). The portion of stream traffic, which is generated by real-time critical applications with hard QoS requirements, is subject to flow/call admission control. Therefore, not all of the offered traffic flows will enter the network. For elastic traffic no CAC is assumed.

2.3.1.5 Domain Design

This last step of the initial network design process is specific to service architectures, which employ central entities such as resource managers or policy servers. In large networks single servers might not be able to cope with the total number of requests. Therefore, the networks are divided into domains, which are then served by their own central entity.

The design of such domains, i.e., the specification of their border as well as the placement of the central unit, is an optimization problem, which is similar to network partitioning or node clustering. In [RG02] for example, resource management domains are designed. The optimization objective is the minimization of the call setup delay introduced by the resource management system, taking into account the maximum request rate per resource manager and the maximum diameter of a domain.

2.3.2 Network Management

Network management comes into play after the network has been designed and set up. Network management as a whole describes the process of accompanying a network's operational state and influencing it from time to time in order to keep up the required QoS. Thus, monitoring functionalities have to be implemented within the network operation processes, which raise alarms whenever the desired QoS cannot be delivered. In these cases, network management decisions have to be made and appropriate countermeasures have to be chosen.

From the great number of network management tasks, we focus on routing optimization and bandwidth allocation issues. These two processes allow a network provider to tune the network without having to change its structure and without having to invest in additional infrastructure. Routing optimization at this point usually aims at improving QoS in networks rather than lowering costs. It is often referred to as being part of *Traffic Engineering* [ACE+02].

2.3.2.1 Routing-based Traffic Engineering

In order to achieve higher QoS, based on a given network infrastructure and a certain traffic load situation, a network service provider might have the possibility to better distribute the traffic in the network. By reducing the link load values in the network, overload situations can be avoided. This can be done relying on conventional routing protocols such as OSPF and EIGRP, or by deploying new technologies such as MPLS (see Chapter 3).

The idea of routing optimization is to find a routing pattern, which achieves the best network QoS for a certain demand matrix (predicted or measured). Based on monitoring events or traffic forecasts, routing is changed from time to time. However, every time routing is reconfigured, affected traffic flows might experience service degradations due to short-term transition phases and routing instabilities. Therefore, if routing is adapted frequently, the impact of rerouting has to be kept as low as possible. The number of rerouted flows or the amount of rerouted traffic needs to be taken into account when computing a new routing pattern. On the other hand, if routing optimization is done only on a very coarse time granularity basis, QoS optimality is the main criterion.

Multi-Service Considerations

In multi-service IP networks two different notions of routing optimization exist, depending on the capabilities of the employed routing technology.

It is conceivable that in future IP networks forwarding will be type-of-service specific. Packets belonging to different traffic flows with equal source and destination addresses might traverse the network along different paths. The routes could be selected by taking into account the type of traffic and its QoS requirements (*QoS routing*) [CNRS98]. From a routing optimization point of view, service-specific routing in multi-service networks is similar to regular routing in networks with only one type of service. Since the individual services are treated independently by the routers, their routing patterns could also be optimized independently. Thus, routing optimization strategies for single services could be applied to each service in a multi-service environment.

If routing is not service specific, rerouting always affects all traffic classes equally. Therefore, routing optimization has to consider the effects of route changes on each of the traffic classes. It could be possible that QoS is improved for one service while it greatly deteriorates for the other since the bandwidth allocation scheme is not appropriate anymore after routing optimization.

2.3.2.2 Bandwidth Adaptation

If traffic flows change and the corresponding service degradation cannot be alleviated by routing adaptation, we can try to change bandwidth allocation parameters. Bandwidth adaptation in this context is not the same as capacity extension, which requires the installation of new bandwidth. It refers to the act of moving bandwidth between different traffic classes by modifying scheduler settings within routers.

The optimization issues of bandwidth adaptation are very similar to the ones of capacity assignment, which is part of the initial network design process. The same methodologies can be used to compute the required bandwidth shares. If in the end it turns out that the links do not provide sufficient capacities, extension planning is inevitable.

2.3.3 Extension Planning

The rapid growth of IP traffic makes it impossible to design a network one time and leave it this way for the future. As traffic increases or new services are to be added, a change of network topology might become necessary.

Extension planning often requires a significant revision of the network infrastructure. It is therefore similar to the initial network design process and contains basically the same subtasks. However, contrary to initial network design, an existing network infrastructure has to be taken into account.

2.4 Conclusion

Network planning requires the solution of many different optimization problems. Some of the tasks are interrelated, while others can be solved in isolation. Two problems, which arise repeatedly at different points throughout the network planning cycle, are routing optimization and capacity assignment. Both are crucial elements of the initial network design process, of traffic engineering, as well as of extension planning.

Chapter 3

Routing Optimization

In this chapter, we consider routing optimization in IP-based networks from the perspective of traffic engineering, i.e., decoupled from any other process such as network dimensioning. Doing so, we differentiate between two notions of routing optimization:

- global routing optimization and
- optimized routing adaptation.

With *global routing optimization* we denote the process of finding the best routing solution irrespective of any pre-existing configuration (global optimum). *Optimized routing adaptation*, on the other hand, considers an initial routing pattern. Based on the given configuration, routing is only slightly adapted in order to keep the negative impact of transition phases low. Thus, we are interested in finding an optimum in the neighborhood of the existing configuration.

In either case, we focus on optimization approaches that employ standard routing protocols and existing technologies for intra-domain application. It is our aim to keep the administrational effort low (i.e., minimize OPEX) and to defer the need for advanced technologies as long as possible. Therefore, the core of our optimization process relies on native IP routing protocols, which are only complemented by more advanced technologies if needed.

We first discuss general goals of routing optimization and define the specific objectives, which we pursue in this Chapter. We then briefly introduce routing technologies in IP networks and provide appropriate abstractions. It is necessary at this point to discuss the underlying principles and the consequences for routing optimization. After giving an overview of related work, routing optimization with OSPF/EIGRP and routing adaptation with MPLS are investigated and our algorithms are presented. In order to concentrate on the matter of routing optimization, single-service scenarios are assumed. However, multi-service extensions to our algorithms can be derived in a straightforward manner. Their implementation is pointed out in the respective paragraphs. The applicability and the quality of our algorithms are discussed by means of several numerical examples.

3.1 Objectives of Routing Optimization

The possible objectives of routing optimization are manifold and there is no unique definition of what is considered an *optimum* routing solution. Assuming that the network infrastructure

is given and that link and possibly node capacities are assigned, two objective categories, which intuitively are similar but which reflect two different perspectives of network operation, are:

- maximization of achievable traffic throughput and
- optimization of quality of service.

In the first case, a certain minimum degree of QoS is required, which all routing solutions have to comply with (e.g., a threshold for average end-to-end delay or packet loss). The routes are then optimized with respect to the amount of traffic that can be transferred over the network while still meeting the QoS requirements. Thus, this approach focuses on traffic increase, which the network might have to cope with in the future.

In the second case, the dominating idea is performance improvement for a certain traffic load. This can be seen as a countermeasure against traffic variations, which have been observed over the past and which have to be taken into account for further operation of the network. As we assume that the traffic matrix can be estimated sufficiently well for a certain time period, we adopt this approach in our work. Thus, we consider routing a means of reacting to increasing and varying traffic loads with the objective of optimizing the achievable network QoS. This objective corresponds to the one stated in Chapter 2.

As also discussed in Chapter 2, different ways exist to quantify QoS in a network. Two categories can be distinguished: transaction-based and utilization-based measures. Since the latter are computationally more efficient, we choose this approach for our optimization processes. We consider explicit transaction-based QoS measures only in forms of optimization constraints. Thus, our optimization objective is the *minimization of the maximum link utilization* in the network. If the maximum cannot be kept below a desired threshold, the problem has to be seen as infeasible. In this case, traffic engineering through route optimization fails, and other methodologies such as capacity extension have to be considered.

3.2 Routing Technologies

The expression *routing* as it is used in the Internet community loosely denotes two distinct processes: computation of routes and packet forwarding. Routing technologies have to provide solutions for following tasks:

- discovery and announcement of network states and link changes,
- computation of routes through the network, and
- forwarding of packets from a source node to the destination.

While packet forwarding is performed by the IP layer, topology discovery and route computation is usually carried out by specific routing protocols. Each router within a homogeneous domain is running the same routing protocol, which makes sure that up-to-date network information is gathered and routes are computed.

Internet routing protocols can be classified according to their regional scope and the way routes are computed: *interior routing protocols* (also referred to as *interior gateway protocols* IGP) are used within autonomous systems, while *exterior routing protocols* provide connectivity between autonomous systems. Routers, which are located on the boundary

between two ASes, have to run both types of protocols. On the one side, they have to represent their own AS in order to advertise its reachability. On the other side, they have to inform interior routers about addresses and prefixes of outside networks. Figure 3.1 gives an overview of the most important IP routing protocols.

In the following paragraphs we briefly introduce the most important interior routing protocols. We focus mainly on the issues of link metrics and route computation. Furthermore, we present an overview of the Multiprotocol Label Switching (MPLS) technology, which is not really a routing protocol but a forwarding technique. Exterior protocols like the outdated Exterior Gateway Protocol (EGP)[Mil84] and the Border Gateway Protocol (BGP)[RL95] are not considered in this context, since they will not play a role in our planning and optimization processes. For a detailed treatment of Internet routing protocols, see for example [Hui00] or [PD00].

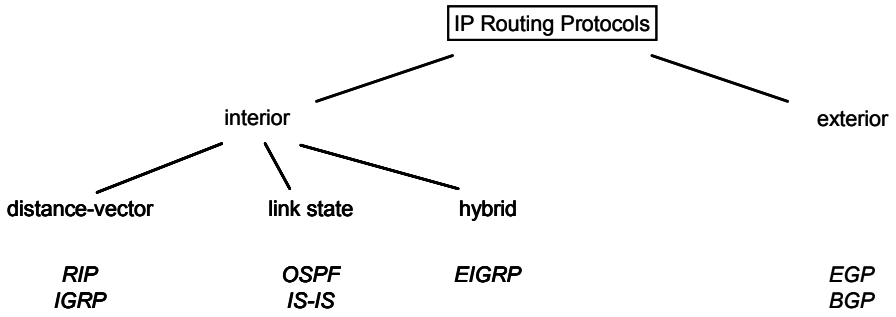


Figure 3.1: Overview of routing protocols

3.2.1 Routing Information Protocol (RIP)

The *Routing Information Protocol* (RIP) belongs to the family of *distance vector protocols*, which are based on the shortest path algorithm by Bellman [Bel57] and its distributed realization described by Ford-Fulkerson [FF62]. RIP was first standardized by the IETF in 1988 [Hed88] and was later extended to become RIP Version 2 [Mal94]. An adapted version for IPv6 is specified in [Mal97].

RIP is a very simple routing protocol, which primarily considers the hop count as the basis for routing decisions. Neighboring routers regularly exchange their routing tables containing the distances, i.e., the number of hops towards all known destinations (therefore the name *distance vector protocol*). From this information each router is able to deduce and update its own distances and again report them to the neighbors. This way, link changes are propagated hop by hop through the network. In order to speed up the distribution process and to alleviate the undesirable *counting to infinity* effect, RIP uses *triggered updates*: routing information is distributed as soon as changes are recognized, not waiting for the next regular update period.

The downside of RIP's simplicity is its inadequacy for large or complex networks. To keep the duration of transient states, which might lead to temporary routing inconsistencies, as low as possible, the maximum allowed hop count is set to 15, with a value of 16 representing infinity (i.e., network unreachable). Thus, large networks cannot be handled by RIP. Furthermore, the small metric limit makes it impracticable to differentiate links respective to their

characteristics such as capacity or propagation delay. The range for assigning different metric values while still allowing a certain network diameter is very narrow.

3.2.2 Interior Gateway Routing Protocol (IGRP)

The *Interior Gateway Routing Protocol* (IGRP) is a Cisco-proprietary distance vector protocol, which was introduced in the late 80's in response to steadily increasing IP network sizes. The main enhancements of IGRP over RIP are a more aggressive loop-avoidance mechanism, a new route computation algorithm, which considers different link attributes, and the ability to perform multi-path routing over unequal-cost paths [Zin02].

At this point, we are mainly interested in the route computation algorithm, as this will play a crucial role later on in this work. With IGRP, every interface (i.e., link) has four different metric types associated with it: *delay*, *bandwidth*, *reliability*, and *load*. The first two parameters are assigned statically, while the third and the fourth are determined by the routers during network operation. When a router computes the shortest path towards a destination, it considers a combination of these metric parameters. The overall path metric function is given by

$$M = \left[K_1 \cdot b + \frac{K_2 \cdot b}{256 - load} + K_3 \cdot d\right] \cdot \frac{K_5}{reliability + K_4}$$

with

$$b = 256 \cdot \frac{10^7}{\min(bandwidth)}$$

and

$$d = 256 \cdot delay$$

The individual components of the metric sum can be interpreted as follows:

- The first term concerns the bottleneck bandwidth ($\min(bandwidth)$), i.e., the smallest link capacity along the path, which is given in kbps. The smaller the bottleneck bandwidth is, the more it contributes to the overall path metric sum. As a consequence it is less likely that a path with a small bottleneck bandwidth is selected for routing.

- In the second term a *load* parameter is introduced, which represents the utilization of the highest loaded link along the path. The granularity is 1/256 and a *load* value of 255 indicates 100% link utilization. This parameter enables adaptive routing where routes are adjusted to current load situations and links with very high loads are avoided. However, it is well known that this kind of routing adaptation might lead to unpredictable and rather undesirable effects such as route flapping and corresponding routing instability [WC92]. Therefore, it is recommended that this option be deactivated by setting the weight factor K_2 to 0.

- The third item of the sum accounts for the total propagation delay, which is the sum of all individual link delays along the path. The link delay values are measured in microseconds and are configured statically by the network administrator. It is noteworthy that the *delay* parameter is the only additive component of the path metric.

- The last term stresses the issue of reliability. The parameter *reliability* corresponds to reliability of the least-reliable link along the path. It is a probabilistic measure, which routers observe during run-time. The higher the reliability of a path is, the more preferable it appears for a router. With the two weight parameters K_4 and K_5 the influence of this metric can be adjusted. By definition, setting $K_5 = 0$ turns off the reliability option.

In default router configurations, which are recommended by Cisco, the weight factors are $K_1 = K_3 = 1$ and $K_2 = K_4 = K_5 = 0$, which leaves us with the simplified path metric formula

$$M_{default} = \left[\frac{10^7}{\min(bandwidth)} + delay \right] \cdot 256$$

As mentioned earlier, IGRP supports multi-path routing even over paths, which do not have exactly the same metric sum. If the difference between the shortest and the next shortest path is smaller than a certain configurable value, the traffic is divided up between the respective outgoing interfaces. The ratio of the traffic shares depends on the deviation of the path lengths.

3.2.3 Open Shortest Path First (OSPF)

Open Shortest Path First (OSPF) was standardized by the IETF in 1998 as a solution for large networks [Moy98]. It belongs to the class of *link state protocols*. Instead of exchanging distance metrics between neighbor routers, all OSPF routers distribute link state information associated with their interfaces to all other routers in the network. This way, every router builds up and maintains its own topology database, which contains elements representing subnets (stub networks), OSPF routers, transit networks (i.e., subnets that connect several routers), aggregated networks (i.e., areas), and destination networks outside the AS. Figure 3.2 illustrates the mapping of an AS into a network graph, whose structure is then stored in an OSPF router's database.

Based on the global view of the network topology, every router is able to perform shortest path computations independently and determine the relevant outgoing interfaces. OSPF allows four different types of metrics being associated with every link: *distance*, *cost*, *delay*, and *reliability*. However, for path computations only one metric is used at a time. Theoretically, it would be possible to compute four different routing tables (one for each metric type) and, then, forward packets according to one of the schemes. However, to guarantee consistency among all routers, this would require that IP packets are marked appropriately and that every router applies the same forwarding scheme. In practice, this feature is not used and routes are computed only for one additive metric type.

OSPF supports multiple paths towards one destination. However, it only allows equal-cost load sharing, which means that traffic destined to a certain destination is split equally over all routes with the same shortest path lengths.

Annotation

The *Intermediate System - Intermediate System* (IS-IS) protocol is the interior routing protocol specified by the International Organization for Standardization (ISO), which can also

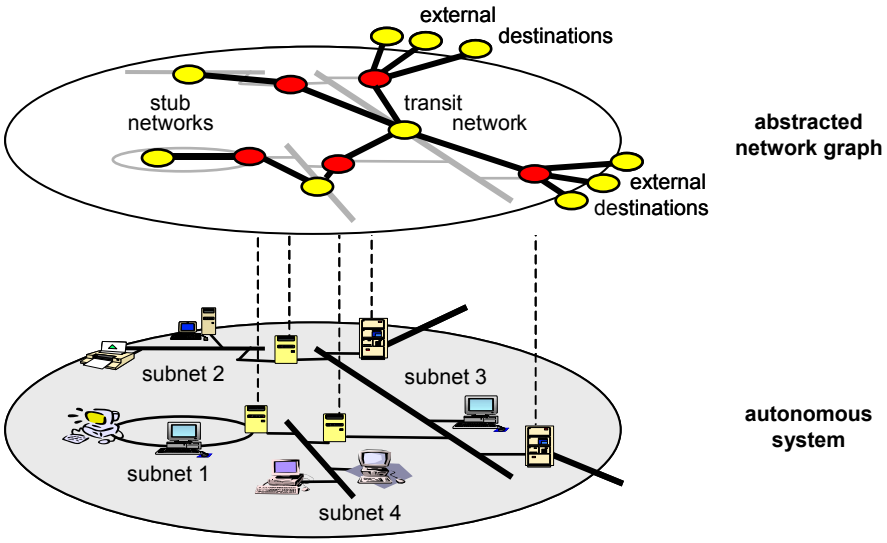


Figure 3.2: Network abstraction of OSPF

be applied in IP networks [Cal90]. Since its principles are very similar to OSPF, we omit the description at this point.

3.2.4 Enhanced Interior Gateway Routing Protocol (EIGRP)

The *Enhanced Interior Gateway Routing Protocol* (EIGRP) uses the same path metric function as IGRP, however, providing more bits for the encoding of delay and bandwidth values (32 bits vs. 24 bits). It also allows equal-cost as well as unequal-cost multi-path routing. The functionality of EIGRP was greatly extended. The most noteworthy feature of EIGRP is the implementation of the *diffusing update algorithm* (DUAL) [GLA93, AGLAB94]. This algorithm assures that routing is consistent (i.e., loop-free) at all times, even when it is in a transient state due to link changes.

3.2.5 Multiprotocol Label Switching (MPLS)

Multiprotocol Label Switching (MPLS) is a technology, which replaces traditional IP packet forwarding by a form of circuit switching [RVC01]. Forwarding decisions are not based on an IP packet's destination address, but on labels, which are added to the packet's header.

The development of MPLS was motivated by two issues:

- to speed up the forwarding process for IP packets and
- to increase routing flexibility.

The first issue concerns the routing-table look-up process when forwarding IP packets. For every packet an IP router has to search its routing table in order to find the network prefix, which best fits the IP packet's destination address. As network prefixes can have different lengths and various prefixes can contain the same IP address, the duration of the look-up process varies. To speed up this process, a fixed-size label is added to the IP packet's

header, which then is taken as the basis of forwarding decisions. However, with increasing performance of current routers, this specific use of MPLS has become rather irrelevant.

The second issue aims at traffic engineering and performance optimization in IP networks where MPLS provides greater routing flexibility (see section 3.3.3).

At the edge of an MPLS network, ingress routers classify incoming IP packets and assign them to specific *Forwarding Equivalence Classes* (FEC). Every packet belonging to a certain FEC is then marked with an appropriate label and sent to the respective neighbor. Within the network, a *label-switching router* (LSR) makes its forwarding decision based on this label and on the number of the interface, on which the packet was received. This way, the IP packet travels along a pre-configured path until it reaches the edge of the network. There, an egress router removes the label and forwards the packet according to standard IP. All IP packets, which enter the network at the same ingress router and which are assigned to the same FEC, traverse the same path through the network. Figure 3.3 illustrates the process.

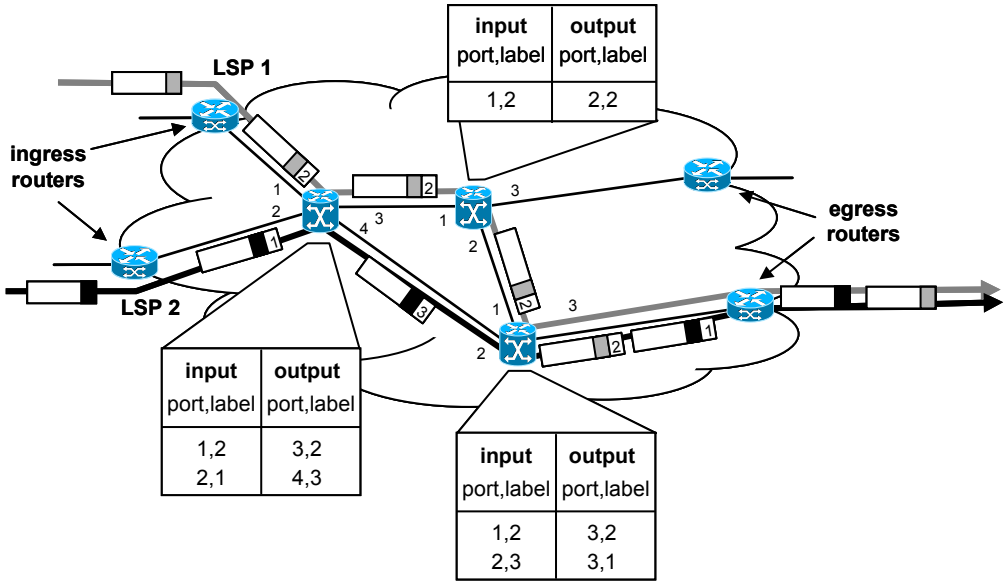


Figure 3.3: Principle of MPLS technology

The validity of a label is limited to one hop (i.e., on the link between two label-switching routers). However, since the forwarding and label swapping scheme is configured before the data phase, IP packets will always travel along a fixed path (*label switched path* LSP). In order to enable several LSP flows to be aggregated, label stacking is possible. This means that several labels can be appended to an IP packet's header and LSRs only consider the first one for forwarding.

The computation of routes and the distribution of label information is not specified in MPLS. Route computation could be done either centrally or in a distributed fashion. A central management server could determine the LSPs and then upload the information onto the LSRs along the way. For the distribution of labels the IETF has specified the label distribution protocol (LDP) [ADF+01]. However, since LDP supports only hop-by-hop specification of LSPs without QoS-capabilities, RSVP-TE [ABG+01] and CR-LDP [JAC+02] were developed. Both protocols enable source routing and can be used in the context of traffic engineering.

3.3 Fundamental Principles and Constraints

After having introduced the technologies of IP routing we can now investigate their underlying principles in view of routing optimization. The particular focus is on route computation and packet forwarding as these two issues determine the constraints of our optimization problem and, thus, the potential of routing optimization. The possible gain in QoS that can be achieved strongly depends on the flexibility of the deployed routing protocol. Therefore, it is necessary to investigate existing routing concepts and discuss their implications for routing optimization.

The way routers update link information has to be considered only when implementing a routing optimization algorithm within a network management framework. However, it does not influence the optimization problem itself.

3.3.1 Metric-based Routing Optimization

The rationale behind routing optimization based on conventional interior routing protocols such as OSPF and EIGRP is that routing in a network can be positively influenced by setting the link weights, i.e., the values of the link metrics, appropriately. As we have seen earlier, IGP protocols select the outgoing interface towards a destination based on a shortest-path computation. This computation takes into account one or more metric values, associated with every link in the network. By configuring every link with the right weights (for example, by using a network management tool), the employed routing protocol can be influenced. The good thing about this form of routing optimization is that it can be carried out without having to apply any special technology. After the weights are set, the conventional routers autonomously propagate the metric information within the network and calculate the routes. No special action has to be taken.

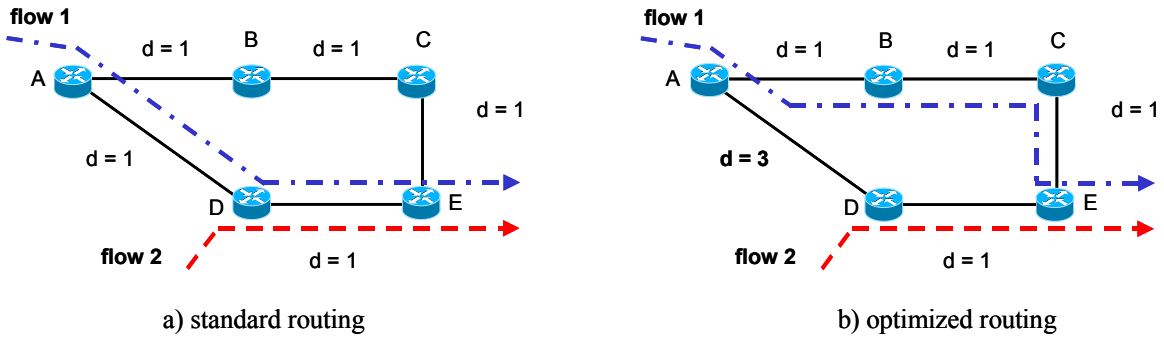


Figure 3.4: Principle of metric-based routing optimization

The idea of metric-based routing optimization is illustrated by means of a simple example in Figure 3.4. In the network on the left hand side, all metrics are additive and set to one. This way, the packets follow a hop-count shortest path from the source to the destination. In the example, this configuration would guide the two IP flows over one common link, possibly creating overload while other links are unused. Changing the weight of link (A-D) from 1 to 3 has the effect that packets of flow 1 are diverted from their original path and traffic load is more evenly balanced across the network. While link metrics often have physically relevant

meanings such as *propagation delay* or *cost*, they can also be used in a generic way purely for the sake of routing optimization. By setting appropriate link weights, one can implicitly influence and, thus, optimize the routing scheme.

3.3.2 Single-Metric vs. Multiple-Metric Route Computation

The differentiation between single-metric and multiple-metric routing concerns native IP routing protocols where routes are determined in a distributed fashion. Since each router computes the outgoing interfaces autonomously, routing consistency can only be guaranteed if all routers have the same information and apply the same algorithm. With MPLS, the routes are often computed at a central instance and are then uploaded (in form of label switching information) onto the label switching routers. Routing consistency is therefore not a big issue when computing the routes.

In OSPF, a link can have four different types of metrics assigned to it. However, for routing computation only one additive metric (such as *delay* or *cost*) is taken into account (*single-metric routing*). In contrary, EIGRP offers the possibility to consider a combination of up to four metric types when computing the shortest-path (*multiple-metric routing*). As introduced earlier, the recommended default configuration activates only two metrics: one additive metric (*delay*) and one concave metric (*bandwidth*). Taking the default configuration, routing preference is given to paths that are shortest in terms of a combination of low total delay and high throughput. Without loss of generality, we normalize the metric function $M_{default}$ and use following expression instead:

$$M_{norm} = \left[K_1 \cdot \frac{1}{\min_l(bw_l)} + K_3 \cdot \sum_l d_l \right] = \left[K_1 \cdot \max_l(icm_l) + K_3 \cdot \sum_l d_l \right]$$

where l are the links along a path. Parameters bw_l and d_l are the properly scaled bandwidth and delay values of the links. Parameter $icm_l = \frac{1}{bw_l}$ denotes the *inverse capacity metric*, which we use instead of the bandwidth. To illustrate the impact of bandwidth-related metric components see Figure 3.5. Link A-D has only a (normalized) capacity of 0.25 and, therefore, contributes to M_{norm} with a reciprocal bandwidth metric of $1/0.25 = 4$. Thus, the cost of path A-D-E is 8, which is the sum of the delay values ($= 4$) plus the bandwidth component, which in this case equals 4. Therefore, router A would choose router B as its next-hop neighbor, where it sees a total metric of only 7. Note that the choice of outgoing interface for router A would be different if only the additive delay values d were considered.

Depending on the scaling of values d and icm, emphasis is either put on small overall delay, on high throughput, or on a mixture of both. In case that delay metrics are substantially larger than inverse capacity metrics, the overall path metric is mainly determined by delay values. Only when there are several alternatives with equal smallest delay sum, the bandwidth component really matters. The router then selects the outgoing interface with the largest possible throughput (*widest-shortest path*). In the opposite case ($icm >> d$), high throughput paths are preferred, and delay is mainly used to break ties (*shortest-widest path*). Whenever the two link metrics are of the same order, no clear preference is given to either one of them. Link metrics are then used in their most generic form as means of routing optimization, without any physically relevant meaning.

Routing optimization based on the multiple-metric concept has some advantages over the single-metric-based shortest-path approach, as can be demonstrated on the network scenario

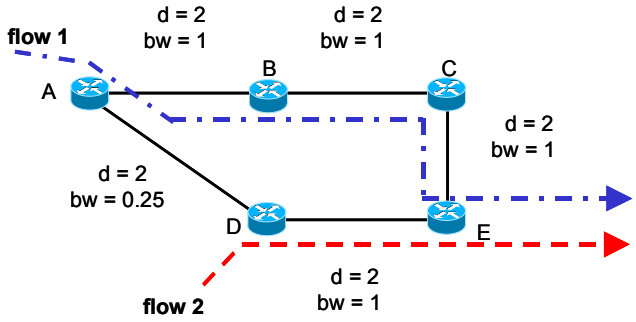


Figure 3.5: Dual-metric routing optimization

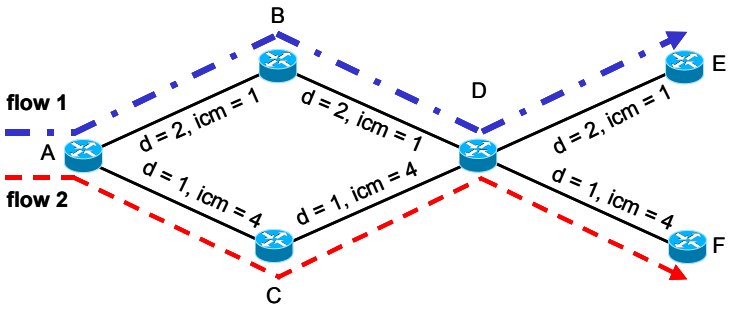


Figure 3.6: Advantage of dual-metric routing optimization

in Figure 3.6. Assume we have two traffic flows with different destinations, whose paths have several nodes in common. Let A be the first node where the two flows come together and D be last common node on their way. While single-metric shortest-path routing would merge the flows at node A and send both of them either over B or over C, multiple-metric routing protocols can achieve the flow pattern given in the figure. For flow 1, the chosen path has a total metric of 7, while the link metrics along the route via C would sum up to 8. For flow 2 the situation is different. The total metrics of the upper and the lower path are 9 and 7, respectively. The trick is to use the inverse capacity metric to make one path option appear more costly for one traffic flow, while for the other flow a larger icm value has no extra effect. It experiences already high icm values on other links along the path, which the two flows do not share.

From this small scenario we can conclude that routing optimization based on dual-metric routing protocols has the potential to outperform its single-metric counterpart as it can realize additional routing patterns. It allows finer granularity of flow treatment in networks with alternative paths. Note that it is always possible for dual-metric routing to produce a solution of single-metric protocols. Setting all $icm = 1$ practically corresponds to single-metric routing.

3.3.3 Destination-based vs. Flow-aware Packet Forwarding

As discussed earlier, native IP routing protocols follow the *next-hop destination-based routing* paradigm. When forwarding a packet, a router determines the outgoing interface based solely on the destination address of the packet and, then, hands it to the respective neighbor router. The neighbor router handles the packet the same way, not caring about where it came from. While being simple and quite efficient, this routing procedure imposes a major limitation on possible path patterns. This is illustrated in Figure 3.7. Whenever two traffic flows with the same destination address cross each others way, they share the rest of the path and run over the same links. Even the use of multiple metric types cannot prevent this. This might create overload on some links, while others are only lightly used or even empty.

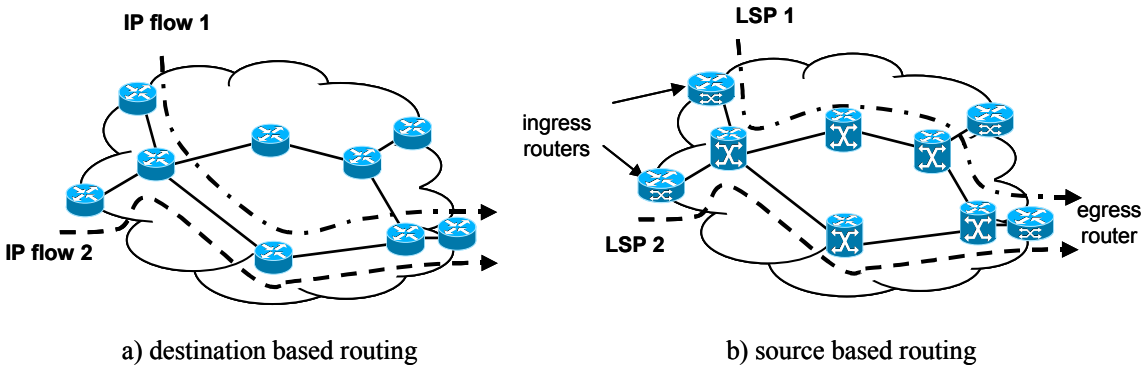


Figure 3.7: Destination-based vs. flow-based routing optimization

To overcome this limitation, the possibility of source-based or flow-aware routing is introduced. Routers base their forwarding decisions not only on the destination address of a packet, but consider some extra information about the packet's path through the network. MPLS makes this possible by allowing dedicated paths from any source node to any other destination node. This introduces a high degree of "routing freedom" as any desired routing pattern can be achieved.

Since MPLS can reproduce all destination-based routing patterns, it can be concluded that routing optimization based on conventional IP routing protocols can never outperform routing optimization with MPLS. However, MPLS requires the installation of additional technology together with extra administrational effort. Native IP routing protocols on the other side are standard.

3.3.4 Hybrid Routing Optimization

Different strategies exist to exploit the routing flexibility and the optimization potential of the MPLS technology in already existing IP networks.

Most often the whole network or at least the backbone area is turned into one or more MPLS domains and all IP packets are subject to label-switching. This approach requires that new technology is installed at each router location. Every router has to support ingress/egress and/or label-switching functionalities. Furthermore, for every ingress and every egress router at least one LSP needs to be established. Assuming fully-meshed single-hop configurations (one LSP for every ingress-egress pair) the total number of LSPs is $N \times (N-1)$, with N

being the number of ingress and egress points. For larger networks this might become an administrational challenge, especially since most LSPs are still set up manually.

In order to reduce CAPEX and OPEX, it is conceivable that only a few routers are extended by MPLS functionality. This way, only a subset of all end-to-end flows is routed along explicit LSPs, while the majority of packets still follows regular IP routes. This *hybrid routing* approach is possible since MPLS does not interfere with IP routing. Actually, MPLS routers have to be able to recognize pure IP packets, since even in homogeneous MPLS domains management and signaling messages are exchanged over regular IP. Taking the example of Figure 3.7, the same routing solution can be achieved if only flow 1 is label-switched, while flow 2 still follows the regular path. In the remainder of this dissertation, we will refer to a flow, which is subject to MPLS label-switching, as a label-switched flow (LSF). Since it is possible that for one end-to-end flow several LSPs are established (MPLS multi-path), one LSF is not necessarily the same as an LSP. For one LSF we can have one or more LSPs.

3.3.5 Multi-Path Routing

A feature of routing protocols, which influences the optimization process, is load sharing. In destination-based routing protocols this capability is often implemented in form of the *equal-cost multi-path* concept (ECMP). Whenever a router can reach a destination node via several paths with equal metric sums, it splits up the traffic evenly across all corresponding outgoing interfaces. Although EIGRP would also support unequal cost multi-path, we do not consider this option in our optimization.

If multiple-metric routing protocols are used, "equal cost" does not necessarily mean that all metric components are the same on all load-sharing paths. This is illustrated in Figure 3.8. While the metric combinations of both paths from B to C are equal, their delay and bandwidth portions are quite different.

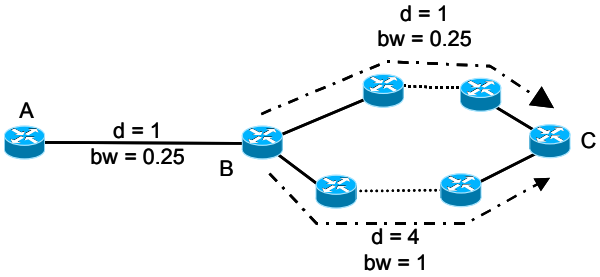


Figure 3.8: Equal-cost paths with dual-metric routing protocols

In order to compute the route from A to C, router A needs to know metrics d and icm of the path from B to C. Therefore, B has to choose one representative interface among the outgoing interfaces, whose metric set is then passed on to the upstream routers. Although both paths have equal costs the choice of metric pair that B reports to A affects the cost value of A towards C. Receiving the metrics of the upper path results in a total cost value of 6, while the lower path metrics give a value of 9. To allow unambiguous route computation we define a *primary outgoing interface*, whose metric values are reported to neighbor routers. In this work, we assume following assignment: Among all outgoing interfaces, the primary interface

is the one with the highest throughput value (i.e., lowest icm) towards the destination. In the given example, this would be the lower path with $icm = bw = 1$.

MPLS can also support multi-path routing. An ingress router can assign incoming packets with the same destination (egress) router to several FECs. Combined with load-sharing capabilities, the label switching technology provides a most flexible tool for traffic engineering, which can implement any desired flow pattern.

3.4 Related Work

Routing optimization in IP networks has received a considerable amount of attention in the recent literature. Due to the currently tense economic situation of many network service providers, a majority of the publications deals with routing optimization as a cost-efficient means of traffic engineering. However, there has also been work done where routing optimization is part of an overall network design problem and, thus, is not considered in isolation.

3.4.1 OSPF-based Traffic Engineering

In [FT00] and [FRT02] Fortz et al. motivate the idea of using OSPF/IS-IS routing with optimized weight settings in order to improve network QoS. The optimization objective is $\sum_{l \in links} f(\rho_l)$, where $f(\rho_l)$ is a piecewise linear and convex function over link utilization ρ_l. As ρ_l approaches 1, the slope of f drastically increases, causing the optimization process to avoid link overload situations if possible. For solving the optimization problem, a guided search heuristic similar to tabu search (however, with hash tables instead of tabu lists) is proposed. This heuristic explores the neighborhood of an initial weight vector by modifying individual link metric settings in a greedy fashion. In case of equal shortest paths, the load is evenly split up (OSPF ECMP). To diversify the search process, random perturbation of link weights is performed after finding a local minimum. In [FT02] further investigations of routing optimization are presented, taking into account failure situations, multiple load periods, as well as noisy traffic matrices. For the considered scenarios, mainly based on AT&T's IP backbone with 90 nodes and 274 links, it can be demonstrated that routing optimization is a suitable means for traffic engineering.

The work of Ericsson et al. [ERP02] considers the same optimization problem with identical objective function as in Fortz's work. However, a different search strategy for the weight setting problem is proposed. Ericsson's approach is based on a genetic algorithm, where individual solutions of the problem are represented by means of the link weights. This application of the genetic algorithm framework corresponds in principle to the one presented in [Rie98]. Resende presents a memetic version of the algorithm, which in addition to the genetic algorithm implements a local search heuristic [Res02].

In [RR01] Ramakrishnan and Rodrigues discuss different forms of shortest-path routing optimization. In this context they introduce the N commodity linear program formulation for the general multicommodity (MC) flow problem [AMO93]. This formulation takes into account the destination-based forwarding behavior of current intra-domain routing protocols. It requires less variables and, thus, is more efficient than the general MC formulation. However, the formulation is not capable of capturing all relevant characteristics of the OSPF protocol. Therefore, a combinatorial search procedure is presented. Starting from an initial network

configuration, the link weights are iteratively modified in order to drive network QoS towards a local optimum. The average network delay (as computed for M/M/1 waiting systems) serves as an objective. The algorithm is applied to a network with 16 nodes and 18 links.

Wang et al. [WWZ01] also approach the routing optimization problem through use of the MC flow formulation. Their primary objective is the minimization of the maximum link utilization, which results in a well-balanced network. They show that any set of optimal routes, which is obtained from the MC linear program, can be represented by shortest path routing where the link weights are determined by the dual problem. However, the presented approach does not capture realistic features of existing protocols such as the need for integer link weights or the incapability of arbitrarily distributing traffic across equal-cost paths.

Further work dealing with routing optimization in the context of traffic engineering and performance improvement has been published by Staehle et al. [SKK00b, KSK02]. The protocols, which the routing optimization procedures rely on, are again conventional IGP protocols with one additive metric. The optimization problem is formulated as a mixed-integer program, which is then solved with CPLEX. As objective function, a weighted sum of the average and the maximum link utilization is considered. Due to the complexity of the problems, which increase with the number of nodes, links, and flows, the approach only works for smaller network sizes. The algorithm was applied to a network with 6 nodes and 7 links. For a larger network (14 nodes, 21 links) the solver could not always find the optimum solution. Heuristic approaches are not presented by the authors. However, for larger networks a separation algorithm is proposed. Large networks are broken down into smaller areas, which then are dealt with individually before being reunited [MKSB02]. The applicability of this approach is demonstrated for networks up to 25 nodes and 42 links.

In [MK02b] and [MK02a], Mulyana and Killat propose a hybrid genetic algorithm for the routing optimization problems. The objective function considers a weighted sum of the average and the maximum link utilization in the network. In a second approach, the maximum link utilization is replaced by the number of modified link weights. This way, weight changes can be taken into account and minimized. However, it does not seem to be possible to set a certain target value for the number of desired changes. Only by varying the weight in the objective function, the emphasis of the optimization can be shifted and, thus, different results can be obtained. The algorithms were applied to networks with up to 29 nodes and 100 links.

3.4.2 OSPF Optimization in the Context of Network Planning

In [BGW98], Bley et al. consider routing optimization as an embedded problem of network planning for IP networks. While the overall objective is the minimization of capacity costs, the final routing pattern has to follow the typical IP next-hop destination-based paradigm. The problem is formulated as a mixed-integer program. Again, only one additive metric is considered. Furthermore, a special feature of the approach is that it assures that shortest paths are unique. Thus, it does not allow taking into account the possibility of load-sharing across equal-cost paths. As the overall problem becomes very complex for even medium-size networks, heuristics are presented for finding an initial topology and for improving it. The improvement process carries out routing optimization where the quality of a solution is determined by its required capacity costs. The heuristic is based on a local search, which

iteratively scans the neighborhood for better solutions. The overall method was applied to a network with 10 nodes and up to 13 links.

More work where routing optimization appears in the context of network planning is presented by Holmberg and Yuan [HY01b, HY01a]. This work extends the mathematical formulation of [BGW98] and introduces the possibility of equal-cost load-sharing paths. The objective function is again the total sum of all cost values in the network plus a penalty component for traffic demand, which cannot be served. The problem is solved by means of a simulated annealing algorithm, which is applied to the set of integer link weights. Given a certain link weight set, the dimensioning result can be easily deduced. The algorithm was applied to networks with up to 24 nodes and 76 links.

3.4.3 MPLS Routing Optimization

Although originally developed as a technology for fast switching, MPLS is now a popular tool for routing optimization in the context of traffic engineering [AMA+99, BGH+02], where explicit paths are set up for all end-to-end traffic demands. As discussed earlier, the flexibility of MPLS allows the implementation of any desired routing pattern - even the optimal one, which can easily be computed by solving a version of the linear multicommodity flow problem. However, this approach might require the installation of more than $N \times (N-1)$ LSPs where N is the number of nodes in the network.

Y. Wang and Z. Wang [WW99] consider the routing optimization problem with the objective of minimizing the maximum link utilization. They propose a heuristic algorithm, which takes the solution of the linear multicommodity flow problem as a starting point. As this solution usually contains demands, which are split over multiple paths, a subsequent algorithm reroutes all bifurcated flows in order to obtain only one LSP per end-to-end demand.

Xiao et al. [XHBN00] discuss MPLS traffic engineering in IP networks in general and in the context of constrained-based routing. They present a simple heuristic for finding appropriate, bandwidth-constrained LSPs for certain end-to-end flows. In order to reduce the number of required LSPs, they propose a hierarchical MPLS configuration. Instead of connecting all routers in a network by individual LSPs, the network is divided into several domains, which are only internally fully-meshed. The individual clusters are sparsely connected by LSPs. However, no method of how to set up the two-level hierarchy is specified.

Schnitter and Haßlinger [SH02, HS03] present a heuristic algorithm for the LSP routing problem, which is based on the simulated annealing framework. Their optimization objective is the minimization of the maximum link utilization in combination with the sum of the individual link loads. By introducing the latter term in the objective function, LSPs are punished, which do not follow the shortest path.

In [KB03] Köhler and Binzenhöfer present a hybrid traffic engineering approach. Based on OSPF routing, individual label switched paths are established in order to improve network QoS. The flows, which are turned into LSPs, are selected by a heuristic. This heuristic primarily considers flows, which are traversing highly utilized links and which have small bandwidth demands. After selecting a set of flows, the routes of the LSPs are determined by solving the linear multicommodity flow problem in a network preloaded with OSPF traffic.

3.4.4 Conclusion

So far, all papers about IGP routing optimization have considered only one additive metric. In our work, we additionally introduce concave link metrics and show that considerable QoS improvements can be achieved. We present a mixed-integer program, which extends the model of [BGW98] and which is capable of capturing the features of OSPF as well as of EIGRP [RS02]. For large networks a genetic algorithm has been developed, whose principles were first presented in [Rie98] and which was enhanced by a search algorithm [Rie02]. A realization for OSPF-based routing adaptation was published in [Rie03].

In the field of MPLS routing optimization, we propose label-switching as a complement to conventional IP routing [Rie03]. In principle, this approach is similar to [KB03]. However, instead of using a heuristic, we formulate a mixed-integer program, which can be solved optimally for most cases. By applying our algorithm to some of the problems in [KB03] a considerable QoS advantage could be demonstrated.

3.5 Routing Optimization with OSPF and EIGRP

In this section, our methodologies for routing optimization based on OSPF and EIGRP are presented. At first, a mixed-integer linear programming model is specified, which allows solving the routing problem optimally for small networks. Although its applicability is somewhat limited, it is a good means to describe the problem unambiguously. Furthermore, as computation power keeps increasing, modern solvers can cope with more and more complex problems. However, for the moment we often have to rely on heuristics, which find good but not necessarily optimal solutions for medium-size and large network. Therefore, we present a framework, which is based on the idea of genetic algorithms.

3.5.1 Mixed-Integer Programming Model

Based on a given network topology with fixed link capacities and a known demand matrix, our traffic engineering model minimizes the maximum link utilization in the network by adjusting the individual delay and inverse capacity metrics. Since these values are originally related to physical link characteristics (propagation delay or link length and capacities) and their liberal use solely for the purpose of traffic engineering might create discomfort, it should be possible to limit the range of adjustment in order to stay as close to the "physically genuine" values as desired. However, one must be aware that this restriction might also limit the traffic engineering potential.

During our traffic engineering procedure we allow link utilization values to be larger than one. If after the optimization the maximum utilization value is still larger than one or greater than any desired threshold, the network has to be regarded as being overloaded and capacities need to be extended. The value of the threshold usually depends on the type of traffic, which is carried in the network. Delivering real-time traffic with appropriate quality of service requires much smaller link utilizations than transporting best-effort data.

The optimization task is stated as a mixed-integer linear problem. We try to follow the routing process and the routers' computational procedure as closely as possible. We believe that this makes it easier to further extend the model in order to consider additional relevant routing features. Specifically, the model should consider:

- destination-based forwarding,

- single as well as dual-metric routing protocols, and
- multipath forwarding over equal-cost paths.

3.5.1.1 Parameters

The IP network is modeled as a directed graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with node set $\mathcal{V}$ and edge set $\mathcal{E}$. Throughout this thesis, we refer to the nodes and the links of a graph by means of their indices, i.e., node v denotes the node with index v, $1 \leq v \leq |\mathcal{V}|$ and edge e denotes the edge with index e, $1 \leq e \leq |\mathcal{E}|$. Furthermore, an edge is often represented through its end points, i.e., edge (i, j) is the edge with source node i and target node j. Every edge $(i, j) \in \mathcal{E}$ has a link capacity C_{ij} associated with it. As mentioned above, it should be possible to limit the range of delay and inverse capacity metrics for each link. Thus, parameters D_{ij}^{min} (≥ 1), D_{ij}^{max} ($\geq D_{ij}^{min}$), ICM_{ij}^{min}, and ICM_{ij}^{max} specify the minima and maxima of the delay and *icm* metrics for link (i, j), respectively.

The traffic demand is represented by a matrix A where $A[u, v]$ is the flow with origin node u and destination node v. We assume that $A[u, v] = 0$ if $u = v$. Let $\hat{A}[v]$ be the total traffic destined to node v, then $\hat{A}[v] = \sum_{u \in \mathcal{V}} A[u, v]$.

Table 3.1 summarizes the parameters.

node set	$\mathcal{V}$
edge set	$\mathcal{E}$
link capacities	C_{ij}
link delay metric limits	D_{ij}^{min}, D_{ij}^{max}
inverse capacity metric limits	ICM_{ij}^{min}, ICM_{ij}^{max}
traffic demand	$A[u, v]$, $\hat{A}[v]$
set of distinct *icm* values	$\{\widetilde{ICM}_1, ..., \widetilde{ICM}_{\widetilde{M}}\}$
incremental *icm* vector	$\mathbf{m}$

Table 3.1: Parameters of mixed-integer program for OSPF/EIGRP routing optimization

3.5.1.2 Optimization Variables

Traffic engineering is carried out by setting the delay and inverse capacity metric values. For every link (i, j) we introduce a delay variable d_{ij} ($D_{ij}^{min} \leq d_{ij} \leq D_{ij}^{max}$) and an inverse capacity metric icm_{ij} ($ICM_{ij}^{min} \leq icm_{ij} \leq ICM_{ij}^{max}$). However, variables icm_{ij} cannot be used just like this in a linear program. Since the solver has to be able to derive the maximum of two *icm* values during the optimization process, it is necessary that they are encoded in a special way. Therefore, we specify a finite set of $\widetilde{M}$ distinct *icm* values $\{\widetilde{ICM}_1, ..., \widetilde{ICM}_{\widetilde{M}}\}$, which are sorted in decreasing order, i.e., $\widetilde{ICM}_1 > \widetilde{ICM}_2 > ... > \widetilde{ICM}_{\widetilde{M}}$, and form an incremental *icm* vector $\mathbf{m} = \{m_1, m_2, \ldots, m_{\widetilde{M}-1}\}$ with $m_n = \widetilde{ICM}_n - \widetilde{ICM}_{n+1}$ for $n = 1, \ldots, \widetilde{M} - 1$. Then, a link's icm_{ij} value can be represented by the metric vector $\mathbf{m}$ and a binary indicator

vector $\mathbf{b}_{ij} = \{b_{ij,1}, b_{ij,2}, .., b_{ij,M}\}$ with $b_{ij,n} \leq b_{ij,n-1}$ for $n = 2, ..., M$ and $M = \widetilde{M} - 1$. We have $icm_{ij} = \widetilde{ICM}_1 - \sum_{n=1}^{M} m_n \cdot b_{ij,n}$ for all $(i,j) \in \mathcal{E}$. During the optimization process, the elements of vectors $\mathbf{b}_{ij}$ are used as variables instead of icm_{ij}. The reason is that this kind of encoding allows the formulation of linear constraints, which specify the maximum of two icm variables (see constraints (3.26) and (3.27) in the next section).

The amount of traffic on a link $(i,j) \in \mathcal{E}$ towards a destination node $u \in \mathcal{V}$ is represented through variables $f_{ij,u}$. This approach takes into account that packets, which are going to the same destination, are treated equally by the routers. A router does not differentiate between the packet's sources. Therefore, it is not necessary to introduce a flow variable for every source-destination pair as it is commonly done in multi-commodity flow problems. Instead, it is sufficient to have a variable only for each destination. This follows the approach presented in [RR01].

In order to consider shortest path routing with delay and icm components, we introduce several *weight* and *potential* variables as well as routing specific *indicator* variables. Some of the variables are just functions of others and are used for the sake of readability. During preprocessing stages they can be substituted. The choice of an outgoing interface at every node i towards a destination u is modeled by the binary variable $t_{ij,u}$. It is equal to 1 if node i uses link (i,j) to forward packets to destination u. Note that for ECMP a router could use several outgoing interfaces to the same destination. Therefore, a node's primary outgoing interface (see Section 3.3.5) is indicated by the binary variable $p_{ij,u}$. To determine the outgoing interfaces, a router has to perform path computations based on a combination of link metrics. We define a node i's *delay weight* $\delta_{ij,u}$ to be equal to the total path delay from node i to node u, in case node i would use node j as its next-hop neighbor towards u. Related to the delay weight, a node i's *delay potential* Δ_{iu} indicates the total delay to node u, which it announces to its neighbors. A node's delay potential is equal to the delay weight associated with the primary outgoing interface. Similar to the delay weight, a node's *capacity weight* $\psi_{ij,u}$ is introduced, that contains the inverse capacity metric, which node i observes along the path to node u if it used node j as the respective next-hop neighbor. The *capacity potential* $\Psi_{i,u}$ of node i holds the relevant inverse capacity metric, which depends on the selected path towards destination node u (again the choice of primary outgoing interface determines the value of $\Psi_{i,u}$). Analog to the inverse capacity metrics icm, capacity weights $\psi_{ij,u}$ and capacity potentials $\Psi_{i,u}$ are represented by the metric vector $\mathbf{m}$ and binary indicator vectors $\boldsymbol{\phi}_{ij,u}$ and $\boldsymbol{\Phi}_{i,u}$, respectively. Vectors $\boldsymbol{\phi}_{ij,u}$ and $\boldsymbol{\Phi}_{i,u}$ have the same structure as $\mathbf{b}_{ij}$ defined above. Thus, $\psi_{ij,u} = \widetilde{ICM}_1 - \sum_{n=1}^{M} m_n \cdot \phi_{ij,u,n}$ and $\Psi_{i,u} = \widetilde{ICM}_1 - \sum_{n=1}^{M} m_n \cdot \Phi_{i,u,n}$. So far, the two routing metric components have only been considered in isolation. Therefore, we bring in one more weight and one more potential variable. A node's *combined-weight* $\omega_{ij,u}$ combines the delay weight and the capacity weight at node i towards node u over link (i,j). The minimum of all weights at a node i towards destination node u is given by the *combined-potential* variable $\Omega_{i,u}$. For a certain metric setting, $\Omega_{i,u}$ is the shortest distance between node i and node u according to the dual-metric function.

Finally, we specify variables ρ_{ij} as well as ρ_{max}, which contain the utilization of the individual links and the maximum thereof, respectively. Table 3.2 gives an overview of the optimization variables.

link delay	d_{ij}
inverse capacity metric	icm_{ij}
binary indicator vector for icm_{ij}	$\mathbf{b}_{ij}$
link flow	$f_{ij,u}$
outgoing interface indicator	$t_{ij,u}$
primary outgoing interface indicator	$p_{ij,u}$
node delay weight	$\delta_{ij,u}$
node delay potential	Δ_{iu}
node capacity weight	$\psi_{ij,u}$
binary capacity weight indicator vector	$\boldsymbol{\phi}_{ij,u}$
node capacity potential	$\Psi_{i,u}$
binary capacity potential indicator vector	$\boldsymbol{\Phi}_{i,u}$
combined-weight	$\omega_{ij,u}$
combined-potential	$\Omega_{i,u}$
link utilization	ρ_{ij}
maximum utilization	ρ_{max}

Table 3.2: Variables of mixed-integer program for OSPF/EIGRP routing optimization

3.5.1.3 Constraints

At first, the limits of the delay and *icm* metrics are specified. For the delay variables this is done with constraints (3.1) and (3.2). For *icm* variables it is more complex. Since *icm* values are encoded through $\mathbf{b}_{ij}$ vectors, we have to translate the limits of icm_{ij} into corresponding restrictions of the elements of indicator vector $\mathbf{b}_{ij}$. Note that an entry of a vector $\mathbf{b}_{ij}$ can only be 1 if all elements to the left of it are also 1. Furthermore, with an increasing number of 1's the represented inverse capacity metric becomes smaller. Thus, for icm_{ij} to stay below an upper bound, one has to fix the $\hat{n}_{ij}$ lowest entries, i.e., variables $b_{ij,1} = b_{ij,2} = ... = b_{ij,\hat{n}_{ij}} = 1$, such that $\widetilde{ICM}_1 - \sum_{n=1}^{\hat{n}_{ij}-1} m_n \geq ICM_{ij}^{max}$ and $\widetilde{ICM}_1 - \sum_{n=1}^{\hat{n}_{ij}} m_n \leq ICM_{ij}^{max}$. Accordingly, for the lower bound the $\check{n}_{ij}$ highest entries have to be set to 0, i.e., $b_{ij,M-\check{n}_{ij}+1} = b_{ij,M-\check{n}_{ij}+2} = ... = b_{ij,M} = 0$ such that $\widetilde{ICM}_1 - \sum_{n=1}^{M-\check{n}_{ij}} m_n \geq ICM_{ij}^{min}$ and $\widetilde{ICM}_1 - \sum_{n=1}^{M-\check{n}_{ij}+1} m_n \leq ICM_{ij}^{min}$. In summary, we have constraints (3.3 – 3.5) where $\hat{n}_{ij}$ and $\check{n}_{ij}$ are set accordingly.

$$d_{ij} \leq D_{ij}^{max} \qquad \forall (i,j) \in \mathcal{E} \tag{3.1}$$

$$d_{ij} \geq D_{ij}^{min} \qquad \forall (i,j) \in \mathcal{E} \tag{3.2}$$

$$b_{ij,n} = 1 \qquad \forall (i,j) \in \mathcal{E},\ n = 1, ..., \hat{n}_{ij} \tag{3.3}$$

$$b_{ij,n} = 0 \qquad \forall (i,j) \in \mathcal{E},\ n = M - \check{n}_{ij} + 1..., M \tag{3.4}$$

$$b_{ij,n} \leq b_{ij,n-1} \qquad \forall (i,j) \in \mathcal{E},\ n = 2, ..., M \tag{3.5}$$

Next, the traffic flow through the network is described. Constraints (3.6), (3.7), and (3.8) are a modified version of the multicommodity flow formulation, which states that at each node the incoming traffic is equal to the outgoing traffic. The constraints take into account the destination-based forwarding principle of Internet routing protocols and describe the conservation law of flows at every node in the network and at the destination nodes. Contrary to the regular multicommodity flow problem, flows are not differentiated respective to their origin. As soon as two flows with the same destination node end up at the same intermediate node, they are merged.

$$\sum_{j:(i,j)\in\mathcal{E}} f_{ij,u} - \sum_{j:(j,i)\in\mathcal{E}} f_{ji,u} = A[i,u] \qquad \forall i \in \mathcal{V},\ \forall u \in \mathcal{V} : u \neq i \tag{3.6}$$

$$\sum_{j:(j,i)\in\mathcal{E}} f_{ji,i} = \hat{A}[i] \qquad \forall i \in \mathcal{V} \tag{3.7}$$

$$\sum_{j:(i,j)\in\mathcal{E}} f_{ij,i} = 0 \qquad \forall i \in \mathcal{V} \tag{3.8}$$

The next constraints relate to the outgoing interfaces of each node and their linkage to the respective flow variables. Equations (3.9) state that node i does not need an outgoing interface if traffic is destined for itself. For all other destinations there has to be at least one outgoing interface (3.10). Constraints (3.11) limit the number of outgoing interfaces per destination to T_{max}.

$$t_{ij,i} = 0 \qquad \forall (i,j) \in \mathcal{E} \tag{3.9}$$

$$\sum_{j:(i,j)\in\mathcal{E}} t_{ij,u} \geq 1 \qquad \forall i \in \mathcal{V}, \forall u \in \mathcal{V} : u \neq i \tag{3.10}$$

$$\sum_{j:(i,j)\in\mathcal{E}} t_{ij,u} \leq T_{max} \qquad \forall i \in \mathcal{V}, \forall u \in \mathcal{V} : u \neq i \tag{3.11}$$

Constraints (3.12) and (3.13) establish the relationship between outgoing interfaces and traffic flow indicator variables. With (3.12), an outgoing interface (i,j) towards u is "activated" whenever a flow with destination u uses link (i,j). Inequality (3.13), which is activated only when $t_{ij,u}$ is one, guarantees that every flow towards u is sent out equally on all outgoing interfaces (i,j) towards destination u whenever it passes through node i. The constant κ_1 needs to be larger than $\max_{i\in\mathcal{V}} \hat{A}[i]$.

$$\kappa_1 \cdot t_{ij,u} \geq f_{ij,u} \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V} \tag{3.12}$$

$$t_{ij,u} + (f_{ik,u} - f_{ij,u})/\kappa_1 \leq 1 \qquad \forall (i,j) \in \mathcal{E}, \forall (i,k) \in \mathcal{E} : k \neq j, \forall u \in \mathcal{V} \tag{3.13}$$

The primary interface is chosen from the set of all outgoing interfaces. Therefore, whenever a link (i,j) is a primary interface it also has to be a regular outgoing interface (3.14). Only one outgoing interface of node i can be the primary one (3.15).

$$t_{ij,u} \geq p_{ij,u} \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V} \tag{3.14}$$

$$\sum_{j:(i,j)\in\mathcal{E}} p_{ij,u} = 1 \qquad \forall i \in \mathcal{V}, \forall u \in \mathcal{V} : u \neq i \tag{3.15}$$

Now we consider the delay portion of the path metric. In (3.16) the delay weight of node i towards u associated with link (i,j) is given by the sum of neighbor j's weight potential and the link delay of (i,j). A node's delay potential is 0 when addressing itself (3.17), and greater than or equal to 1 otherwise (3.18).

$$\delta_{ij,u} = \Delta_{j,u} + d_{ij} \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V} \tag{3.16}$$

$$\Delta_{i,i} = 0 \qquad \forall i \in \mathcal{V} \tag{3.17}$$

$$\Delta_{i,j} \geq 1 \qquad \forall i \in \mathcal{V}, \forall j \in \mathcal{V} : i \neq j \tag{3.18}$$

The next two inequalities make sure that the delay weight of the primary outgoing interface is picked as the delay potential for node i towards u. Constraints (3.19) and (3.20) reduce to $\Delta_{i,u} = \delta_{ij,u}$ if $p_{ij,u} = 1$. Otherwise, they have no effect if κ_2 is only large enough ($\kappa_2 \geq \sum_{(i,j)\in\mathcal{E}} D_{ij}^{max}$).

$$p_{ij,u} + (\Delta_{i,u} - \delta_{ij,u})/\kappa_2 \leq 1 \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V},\ \kappa_2\ const \tag{3.19}$$

$$p_{ij,u} + (\delta_{ij,u} - \Delta_{i,u})/\kappa_2 \leq 1 \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V},\ \kappa_2\ const \tag{3.20}$$

Constraints (3.21) through (3.25) define capacity weights and potentials, and the special form of their respective indicator variables.

$$\psi_{ij,u} = \widetilde{ICM}_1 - \sum_{n=1}^{M} m_n \cdot \phi_{ij,u,n} \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V} \tag{3.21}$$

$$\phi_{ij,u,n} \leq \phi_{ij,u,n-1} \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V}, n = 2, ..., M \tag{3.22}$$

$$\Psi_{i,u} = \widetilde{ICM}_1 - \sum_{n=1}^{M} m_n \cdot \Phi_{i,u,n} \qquad \forall i \in \mathcal{V}, \forall u \in \mathcal{V} \tag{3.23}$$

$$\Phi_{i,u,n} \leq \Phi_{i,u,n-1} \qquad \forall i \in \mathcal{V}, \forall u \in \mathcal{V}, n = 2, ..., M \tag{3.24}$$

$$\Phi_{i,i,n} = 1 \qquad \forall i \in \mathcal{V}, n = 1, ..., M \tag{3.25}$$

The capacity weight of an outgoing interface (i,j) at node i towards destination u is the maximum of the predecessor's capacity potential and the link's inverse capacity metric icm_{ij}. This is the same as an "AND" operation (expressed through (3.26) and (3.27)) on the individual entries of the indicator variables.

$$\phi_{ij,u,n} \leq 0.5 \cdot (b_{ij,n} + \Phi_{j,u,n}) \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V}, n = 1..., M \tag{3.26}$$

$$\phi_{ij,u,n} \geq b_{ij,n} + \Phi_{j,u,n} - 1 \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V}, n = 1..., M \tag{3.27}$$

The primary interface is the link among all outgoing interfaces, which has the lowest capacity weight on the path towards the destination. As explained in the preceding section, this is important in cases where multi-path load-sharing is applied and the different paths have different icm values. Inequalities (3.28) force the capacity weight of the primary outgoing interface ($p_{ij,u} = 1$) to be smaller than or at most equal to the capacity weights of all other outgoing interfaces ($t_{ik,u} = 1$). The capacity potential of node i towards destination u is equal to the capacity weight of the primary interface ((3.29) and (3.30)).

$$p_{ij,u} + t_{ik,u} + \phi_{ij,u,n} - \phi_{ik,u,n} \leq 2 \qquad \begin{array}{l} \forall (i,j) \in \mathcal{E}, \forall (i,k) \in \mathcal{E} : j \neq k, \\ \forall u \in \mathcal{V},\ n = 1, ..., M \end{array} \tag{3.28}$$

$$p_{ij,u} + \Phi_{i,u,n} - \phi_{ij,u,n} \leq 1 \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V}, n = 1, ..., M \tag{3.29}$$

$$p_{ij,u} + \phi_{ij,u,n} - \Phi_{i,u,n} \leq 1 \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V}, n = 1, ..., M \tag{3.30}$$

Finally, we combine the two metric components and look at the combined-weight and combined-potential variables. First, the combined-weight is defined as the sum of a node's delay weight and capacity weight (3.31). If link (i,j) is an outgoing interface towards u, its combined-weight has to be smaller than or at most equal to any other adjacency's combined-weight (3.32) depending on whether this adjacency is an outgoing interface itself or not. The combined-potential of a node i is the minimum over all combined-weight variables at this node (3.33). In case (i,j) is an outgoing interface ($t_{ij,u} = 1$), the combined-potential is equal to this link's combined-weight ((3.34) and (3.35)). For all other adjacent links with $t_{ij,u} = 0$, the respective combined-weight has to be larger (at least by one) than the node's combined-potential (3.35). The constant κ_3 has to be large enough, i.e., ($\kappa_3 \geq \max_{(i,j)\in\mathcal{E}} ICM_{ij}^{max} + \sum_{(i,j)\in\mathcal{E}} D_{ij}^{max}$).

$$\omega_{ij,u} = \delta_{ij,u} + \psi_{ij,u} \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V} \tag{3.31}$$

$$t_{ij,u} + (\omega_{ij,u} + 1 - t_{ik,u} - \omega_{ik,u})/\kappa_3 \leq 1 \qquad \begin{array}{l} \forall (i,j) \in \mathcal{E}, \forall (i,k) \in \mathcal{E} : j \neq k, \\ \forall u \in \mathcal{V},\ \kappa_3\ const \end{array} \tag{3.32}$$

$$\Omega_{i,u} \leq \omega_{ij,u} \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V} \tag{3.33}$$

$$t_{ij,u} + (\omega_{ij,u} - \Omega_{i,u})/\kappa_3 \leq 1 \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V},\ \kappa_3\ const \tag{3.34}$$

$$t_{ij,u} + (\omega_{ij,u} - \Omega_{i,u}) \geq 1 \qquad \forall (i,j) \in \mathcal{E}, \forall u \in \mathcal{V} \tag{3.35}$$

This leaves us with the specification of the link utilization values and their maximum, formulated through (3.36) and (3.37). Since ρ_{max} is minimized in the objective function, it will always be equal to and never larger than the maximum utilization.

$$\rho_{ij} = \sum_{u \in \mathcal{V}} \frac{f_{ij,u}}{C_{ij}} \qquad \forall (i,j) \in \mathcal{E} \tag{3.36}$$

$$\rho_{max} \geq \rho_{ij} \qquad \forall (i,j) \in \mathcal{E} \tag{3.37}$$

3.5.1.4 Objective Function

As it is the aim of our optimization procedure to minimize the maximum utilization on any link in the network, we define following objective function:

$$v = \min (\rho_{max}) \tag{3.38}$$

3.5.1.5 Characteristics and Extensions

The routing optimization problem is $\mathcal{NP}$ hard [FT00]. The number of variables is in the order of $|\mathcal{E}| \cdot |\mathcal{V}|^2$. The number of constraints is roughly $|\mathcal{E}|^2 \cdot |\mathcal{V}|^2$ for M relatively small.

Lower Bound

A lower bound of the problem can be computed by a modified version of the multicommodity flow problem (MC), which takes into account the destination-based forwarding approach of Internet routing. Taking only constraints (3.6),(3.7),(3.8), and (3.37), results in a linear model, which can be solved efficiently.

Routing optimization with OSPF and EIGRP can never achieve a maximum utilization, which would be better than the result of the linear MC problem. In most cases, the final results lie above the lower bound.

Variations of the Model

It is very easy to extend the model in order to consider a weighted metric function with different emphasis on delay and *icm* values. Taking the two parameters K_1 and K_3 of the EIGRP protocol (see section 3.2.2), they can directly be inserted into equations (3.31) in front of the appropriate variables.

It is also straightforward to have the routing optimization model take into account only additive metrics. Introducing the aforementioned parameter K_1 and setting it to 0 can achieve this. However, in this case it would be better to eliminate all *icm* related variables and constraints and replace the combined-weight and combined-potential variables in (3.32) to (3.35) with their delay-specific counterparts. The OSPF optimization model is a subset of the here presented model.

Furthermore, other objective functions are conceivable. If it is desired to consider a tradeoff between the total load in the network and the utilization on the most utilized link, the objective function can be extended in the following way:

$$v_{ext} = \min \left(\rho_{max} + w \cdot \sum_{(i,j) \in \mathcal{E}} \rho_{ij} \right) \tag{3.39}$$

The two individual objectives are combined in a weighted sum. With increasing w more and more flows are forced to use the shortest paths. Consequently, the number of flows taking longer routes in order to decrease the maximum link utilization is reduced.

3.5.1.6 Multi-Service Extension

The optimization model considers only one type of service. Nevertheless, it can still be applied to multi-service scenarios if

- type-of-service routing is employed, or
- priority scheduling is used to provide differentiated QoS and no thresholds for service-specific link utilization values exist.

In the first case, routing optimization can be carried out independently for each service class. In the second case, it is sufficient to focus on overall link loads since lower link utilization values still correspond to higher QoS.

However, if service-specific link load thresholds exist (e.g., due to strict bandwidth allocation for each traffic class), routing optimization needs to differentiate between the traffic classes. Otherwise, one could end up with a routing solution, which might nicely balance the overall traffic load, but which does not adhere to the allocated bandwidth shares and, therefore, causes congestion and overload for some of traffic classes.

To enable service differentiated routing optimization, our model requires only a few modifications. Assume that set $\mathcal{S}$ contains the indices of the traffic classes. If bandwidth partitioning is applied, capacities C_{ij} are turned into service-specific bandwidth shares C^s_{ij} with $s \in \mathcal{S}$. In the same way, traffic parameters $A[u,v]$ and $\hat{A}[v]$ as well as flow and utilization variables $f_{ij,u}$ and ρ_{ij} are subscript over all services, referring now to traffic parameters and variables associated with individual traffic classes ($A^s[u,v]$, $\hat{A}^s[v]$, $f^s_{ij,u}$, ρ^s_{ij}). Constraints (3.6), (3.7), (3.8), (3.36), and (3.37) need to be adapted accordingly so they additionally iterate over all traffic classes $s \in \mathcal{S}$. No further parameters, variables, or constraints have to be changed.

With objective function (3.38) the multi-service model minimizes the maximum service-specific bandwidth-share utilization in the network. Routing is computed in a way that traffic is equally balanced for all traffic classes within their allocated bandwidth shares. However, other meaningful objective functions and also variations of the model are conceivable. Examples are:

- Minimization of the maximum bandwidth-share utilization for only one traffic class, while the bandwidth-share utilizations of all other classes have to stay below a certain threshold. This approach requires that constraints (3.37) only apply to the one traffic class, whose utilization values should be minimized. For all other classes, ρ_{max} is replaced by a service-specific threshold parameter.

- Minimization of the maximum link utilization (over all traffic classes), while some service-specific link utilizations stay below a certain threshold. This means that constraints (3.36) now contain service-independent capacity parameters C_{ij} as in the original model (however, with service-specific utilization and flow variables). Furthermore, the right hand side of constraints (3.37) is replaced by the sum of the service-specific link utilization values, i.e., the total link utilization.
  This variation of the multi-service model is most appropriate for networks, which employ priority scheduling instead of bandwidth partitioning. In these networks, the link load of the higher priority traffic should not exceed a certain threshold (e.g., 10%) in order to achieve high QoS. By minimizing the overall utilization, it can be guaranteed that enough bandwidth resources are left on every link to carry the lower-priority traffic.

3.5.2 Hybrid Genetic Algorithm (HGA)

As mentioned above, the mixed-integer program can only be solved for smaller networks. Therefore, a heuristic algorithm was developed, which can be applied to networks of larger size. It is based on the concept of genetic algorithms (GA). In order to enhance the performance of the GA framework, our algorithm is complemented by a simple search heuristic. In the literature, this combination is referred to as *Hybrid Genetic Algorithm* (HGA) or *Memetic Algorithm* (MA).

In the following paragraphs the key concepts of the hybrid genetic algorithm are presented. We discuss the implementation and show how the HGA is applied to the problem at hand.

3.5.2.1 Fundamentals of Genetic Algorithms

Genetic algorithms were first developed by John Holland [Hol75] and are often used as search and optimization methods [Gol89]. Given a large solution space, one would like to pick out the point, which optimizes an object function while still fulfilling a set of constraints. In network planning, a solution point could be a specific link topology, a routing path structure, or a capacity allocation scheme.

Genetic algorithms are based on the idea of natural selection. It is suggested that an individual's strength to survive in the world is determined by its gene structure and that over many generations only "good" genes prevail, whereas "bad" ones are rejected. Furthermore, it is expected that bringing together individuals with good gene combinations produces again good or even better ones.

Possible solutions are encoded as strings, which can by of any type (e.g., binaries, numbers, characters). The only requirement is that a string can be transformed into a valid solution. With every string a fitness value is associated, which quantifies the quality of the solution. Then, operations of "natural selection" are performed on these strings: reproduction, crossover, and mutation.

The *reproduction* process creates a new generation. Starting from an existing generation, strings are reproduced with a probability respective to their fitness value. Strings, which represent solutions with good properties, have a higher chance to survive than strings depicting solution points with bad characteristics. This principle is also known as *survival of the fittest*.

The *crossover* operator exchanges genetic information between two strings. The strings of two randomly selected solutions are broken up at an also randomly chosen position, and

parts of the strings are exchanged. One hopes that two solutions with good properties create an even better one.

New genetic material is introduced by the *mutation* operator. The values of individual genes are changed and, hence, new solutions are chosen. Mutation becomes important when after some generations the number of different strings decreases because strong individuals start dominating. In a situation of strong dominance of a few strings, the crossover operator alone would not bring any changes and the search for an optimal solution would be ended. To partially shift the search to new locations in the solution space, the mutation operator randomly alters genes.

The flow chart in Figure 3.9 shows the sequence of the basic operators used in genetic algorithms. We start out with a randomly selected first generation. Every string in this generation is evaluated according to its quality, and a fitness value is assigned. Next, a new generation is produced by applying the reproduction operator. Pairs of strings of the new generation are selected and crossover is performed. With a certain probability, genes are mutated before all solutions are again evaluated. This procedure is repeated until a maximum number of generations is reached. While doing this, the all time best solution is stored and returned at the end.

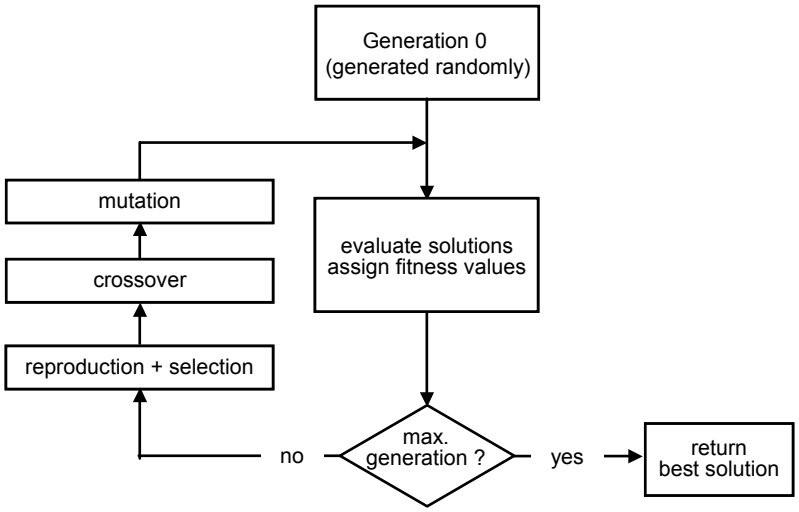


Figure 3.9: Principle of genetic algorithms

3.5.2.2 String Representation

At the very heart of every genetic algorithm lies its encoding methodology, i.e., the representation of solutions as strings of genes. The structure of the strings has to allow the representation of every potential solution. Furthermore, it has to be possible that strings, which are modified during the crossover and mutation phase, are again valid solutions or can at least be turned into such through a clearly specified procedure.

In our implementation we choose a representation, which directly reflects the link metrics in the network. All links are enumerated and an integer link weight is associated with each of them. Thus, a specific gene string contains the weights of all links in the order of the

link's enumeration. Based on these weights, the corresponding routing can be computed by applying the respective route computation algorithm. For the single-metric case, one string contains the additive metric of every link. For dual-metric routing, we require two strings, one for the delay and one for the *icm* values.

The individual genes are integers ranging from 1 to a maximum value W_{max}. In practice, routing protocols allow quite large maximum values (e.g., in case of OSPF up to 65535). However, for traffic engineering and routing optimization purposes the metric values can be kept much smaller. In our network scenarios we usually assume a maximum metric value of 20.

Initial Population

The initial population is the first set of gene strings, from which the genetic algorithm originates. In most cases, this set is chosen randomly. However, it is also possible to inject certain solutions by presetting some of the strings accordingly. This way, the algorithm can be influenced and forced to start searching in certain areas of the solution space. Possible starting points could be for example a previously found optimum or the hop-based shortest-path routing solution where every link weight is set to 1.

However, it has been observed that if at the beginning a solution is injected, which is greatly superior over other randomly selected strings, the algorithm often gets stuck there and does not find a better solution. The reason for this is that the injected string would have such a high fitness value that other strings hardly have a chance to survive at all. Within only a few generations, the randomly selected gene strings would die out and only the initially injected solution would be left over. Therefore, presetting strings should be done with care, and it should only be one trial out of many.

3.5.2.3 Fitness Function and Power Scaling

As we would like to minimize the maximum link utilization in the network, we choose our fitness function to be inverse proportional to this value. This way, routing solutions with smaller maximum link utilization receive higher fitness values and, thus, have a higher chance to be reproduced when setting up a new generation.

To adjust the selection process of the genetic algorithm, we apply power scaling to the fitness function:

$$fitness = \left(\frac{1}{\rho_{max}}\right)^{p} \quad , \; p > 0 \tag{3.40}$$

With $p < 1$ we can achieve that fitness values of bad solutions increase relatively to the best ones. Thus, weak solutions have a higher chance to survive, and the dominance and the reproduction speed of superior solutions is reduced. Setting $p > 1$ has the adversary effect. Strong solutions are given even higher weight, while poor solutions die out faster.

3.5.2.4 Reproduction and Crossover

Reproduction describes the transition process from one generation to the next one. In our work, we adopt the roulette-wheel type selection method with slot sizes that correspond to the fitness values [Gol89]. At first, the fitness values of the gene strings are scaled appropriately so their cumulative sum is one. Then, a random generator with uniform distribution is used

to return a number between 0 and 1. Being mapped onto the roulette wheel, this number indicates the string, which is carried over to the next generation. This procedure is repeated until the new generation is complete.

To improve reproduction results, not all individuals are selected randomly. In each cycle, we inject the all-time and the currently best solutions into every new generation. This way it is guaranteed that the search process keeps looking for better solutions around the so-far best solutions.

For the crossover process, two strings are selected randomly. Another random generator with a uniform distribution between 0 and the length of the strings determines the location of where to break up the two strings. Then, their substrings are exchanged.

3.5.2.5 Mutation

With a certain probability p_m individual genes are mutated. In our case, each gene is considered for mutation and it is also possible that one gene string experiences mutation of several genes. The example in Figure 3.10 shows the dependency between the results (maximum utilization) and the mutation probability for network scenario *N11* (see Appendix A). The graph shows the mean and the minimum ρ_{max} over five trials. The lower the average, the better we consider the overall performance of the algorithm. Since genetic algorithms do not guarantee to find the optimum, a lower average indicates that better solutions are found more often. However, the average does not necessarily say anything about the absolute minimum.

It can be noticed that the mutation probability influences the success of the genetic algorithm. Therefore, we always check several parameter settings when running genetic optimization algorithms.

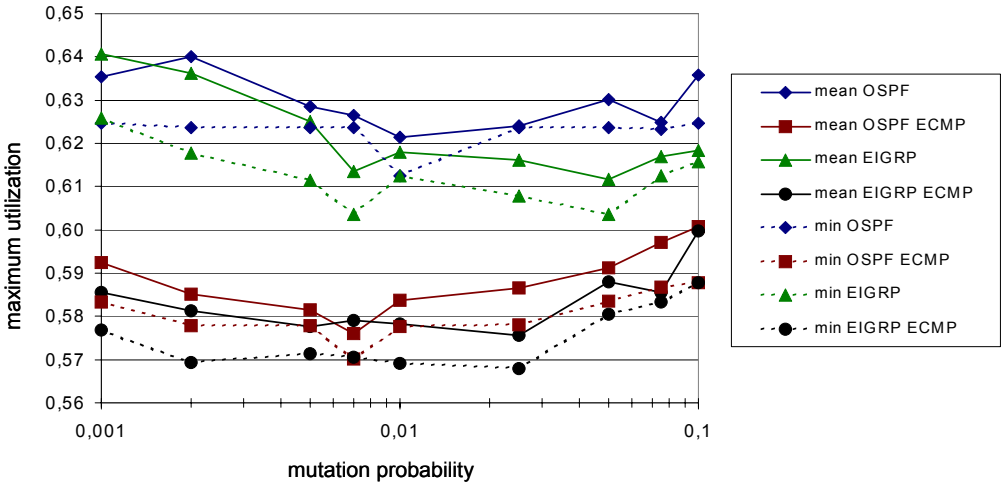


Figure 3.10: Influence of mutation probability

3.5.2.6 Local Search Heuristic

In order to improve the performance and the speed of the genetic algorithm, the evaluation step, which returns the fitness of a solution, is preceded by a local search process. Before

evaluating each routing solution, simple heuristics are used to divert traffic from the link with the highest utilization. This is done repeatedly until no further improvement can be achieved. As the search heuristic is deterministic, a certain metric string always results in the same routing solution.

Single-Metric Heuristic

In case of shortest-path routing with one additive metric, traffic can be diverted from a link by increasing its weight. Therefore, we increment the weight of the highest-utilized link and recompute the routing. If this step leads to a higher utilization value in the network it is reverted and the local optimization heuristic ends. Otherwise, the procedure is repeated for the link, which now has the highest utilization.

Dual-Metric Case

For routing protocols with delay and *icm* metrics there are several possibilities of diverting traffic from individual links. We can increase either the delay metric or the *icm* metric. Furthermore, combinations are possible that might also result in a change of the path structure: the delay metric can be increased while at the same time the *icm* metric is decreased, or vice versa. Thus, the search heuristic iteratively applies these metric modifications to the highest-utilized links. Each metric change is accepted if it does not lead to an increase of the maximum utilization. If no further improvements can be obtained, the heuristic stops.

In all cases it is guaranteed that the run time of the local search process is bounded. For single-metric routing this is obvious since the algorithm only increments link metrics. As the link metrics cannot exceed the maximum metric threshold M_{max}, the local search algorithm is also bounded. For the dual-metric case, guaranteeing a bounded run time is somewhat more tricky. As long as we only increase metric values, we do not run into problems. However, since two of the four metric modification processes decrease one link metric, we have to assure that the algorithm does not get caught in infinite loops. Therefore, we record the states, which have been assumed during the search process. If the process returns to an already visited state, it is interrupted. Thus, a limited run time is guaranteed.

3.5.2.7 Advantage of HGA over GA

The benefits of the local search heuristic are demonstrated in the graph of Figure 3.11. For OSPF routing optimization in the *N40* network (see Appendix A), the best maximum utilization value of each generation is shown over time for several runs of the pure genetic algorithm (GA) and of the hybrid version (HGA). The total time is 300 seconds on a Linux PC with Pentium 4 processor. For the GA the population size is 120, while for the HGA a size of 20 is sufficient. After 300 seconds, the GA has produced about 600 generations, i.e., it has evaluated around 72000 individual strings. During the same time, the HGA has created 900 generations with a total number of 18000 strings. It is noteworthy, that although the population size of the HGA is only a third of the GA's population size, it returns better solutions from the beginning on. Therefore, the results, which are presented in the remainder of this Chapter, were produced with the hybrid genetic algorithm.

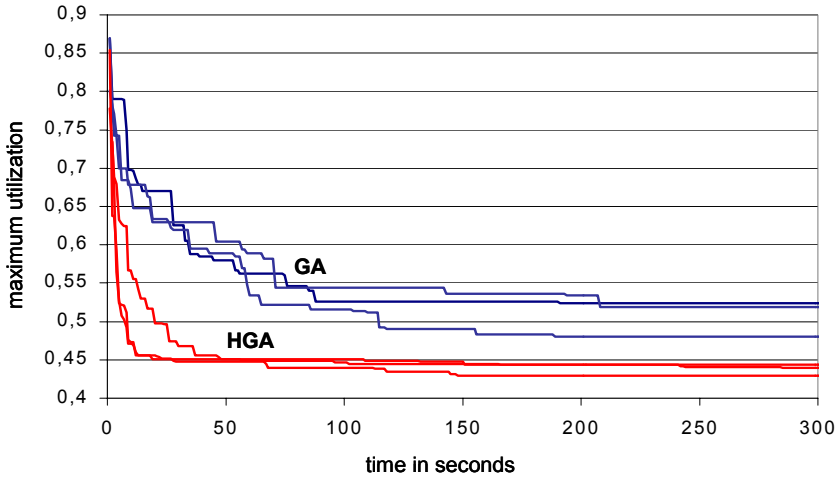


Figure 3.11: Comparison of HGA and GA

3.5.3 Routing Adaptation Extension

So far, we have concentrated on finding the global optimum for routing optimization with OSPF and EIGRP. This might require the change of a large number of link metrics in the network, affecting many flows. In some cases, however, it might be desired to only adjust a given routing configuration and leave most of the flows untouched. This means that the currently active routing has to be taken into account and a tradeoff between the achievable network QoS and the routing impact should be considered.

3.5.3.1 Objectives and Constraints

Two different notions of routing adaptation exist, which both are implemented in our network optimization tool.

First, adaptation could be interpreted as adjusting only a few link metrics. This is motivated by the fact that individual metric changes lead to transition periods with possible routing instabilities. Therefore, if the configuration is adjusted metric by metric, it is best to modify as few metric values as possible. However, the number of metric modifications does not say anything about the number of flows, which are affected by reconfiguration of these link metrics.

Therefore, the second approach considers the number of rerouted flows irrespective of the number of modified link metrics. Assuming that all routers in the network can be informed of link metric modifications at the same time, they all could change their routing appropriately. Therefore, only the flows, which are affected by route changes, experience possible transition effects. Under these conditions, it seems advisable to minimize the number of rerouted flows or the amount of rerouted traffic in order to keep temporary service degradation low. For our optimization procedure, we select the number of rerouted flows n_{flow} as the relevant objective. This way, the results of OSPF/EIGRP-based routing adaptation can be better compared with results that we obtain with MPLS as the number of rerouted flows would correspond to the number of established label-switched paths (see Section 3.7.3). Nevertheless, the amount of

rerouted traffic could be used in just the same way.

In order to realize routing adaptation, the optimization procedure needs to take into account the number of modified metrics n_{metric} and the number of rerouted flows n_{flow}. While ρ_{max} is minimized, the number of modified metrics or the number of rerouted flows should be kept small, possibly below certain thresholds M_{max} and F_{max}, respectively. We can achieve this by adding the appropriate constraint:

$$n_{metric} \leq M_{max}$$

$$n_{flow} \leq F_{max}$$

3.5.3.2 Optimization Procedure

The mixed-integer program of section 3.5.1 is not directly applicable to either one of the two notions of routing adaptation. Due to the flow-based approach it is especially difficult to determine the number of rerouted flows during the optimization process. It would require many new binary indicator variables, which would increase the complexity of the problem even more. Therefore, we do not consider this approach here.

Unlike the mathematical optimization model, the genetic algorithm can easily be extended for routing adaptation with OSPF and EIGRP. To do so, we modify the fitness function (3.40). Instead of ρ_{max}, a weighted sum z is used where penalty terms are added if the above-mentioned constraints are violated. Thus, we have

$$fitness = \left(\frac{1}{z}\right)^p \quad , \; p > 0$$

with

$$z = \rho_{max} + w_{metric} \cdot \hat{n}_{metric}^{k_1} + w_{flow} \cdot \hat{n}_{flow}^{k_2} \tag{3.41}$$

$$\hat{n}_{metric} = \begin{cases} n_{metric} - M_{max} & if \; n_{metric} \geq M_{max} \\ 0 & otherwise \end{cases} \tag{3.42}$$

$$\hat{n}_{flow} = \begin{cases} n_{flow} - F_{max} & if \; n_{flow} \geq F_{max} \\ 0 & otherwise \end{cases} \tag{3.43}$$

By varying the weights w_{metric} and w_{flow} and the exponents k_1 and k_2 of the penalty terms we can adjust the effectiveness, with which the constraints are enforced. For larger values the constraints are strictly enforced. For smaller values, the algorithm might still return solutions where the thresholds are exceeded as long as it helps to reduce the maximum utilization. Setting a weight to 0 deactivates the respective constraint totally.

3.5.4 Multi-Service Considerations

The genetic algorithm can be easily extended so it takes into account multiple services. Only the evaluation step has to be modified. After the routing pattern is determined based on a set of link metrics, the service-specific link or bandwidth-share utilizations are computed. For the fitness function (3.40) the maximum service-specific bandwidth-share utilization could be used. This corresponds to the first approach presented in section 3.5.1.6.

It is also possible that the objective function considers only one traffic class, while the utilization values of other classes need to stay below a certain threshold (also discussed in section 3.5.1.6). In this case, the fitness function has to include the maximum utilization for the one traffic class that needs to be considered, and, additionally, penalty terms for every other service-specific utilization value, which exceeds the respective threshold. This methodology with penalty terms is similar to the approach in the preceding section, where penalty terms are used to keep the number of metric changes and rerouted flows below a threshold.

3.5.5 A Note about Uniqueness of Shortest Paths

The question of uniqueness of shortest paths arises whenever routers are configured to use only one single shortest path towards a destination. In meshed networks with metric-based shortest-path routing it is quite common that several equal-cost paths exist between a source and a destination node. If in these cases the ECMP feature is not activated, it is up to a router to select the relevant outgoing interface. Since routing follows the next-hop destination-based paradigm this ambiguity at individual routers does not create consistency problems concerning routing. Other routers in the network do not have to know, which one of the equal-cost paths is finally chosen as long as the selected path belongs to the set of shortest paths towards the destination. However, for the routing optimization process this is different. If a certain metric setting creates several paths with equal metric sums, it has to be known, which one the corresponding router will select. Otherwise, the final routing pattern could differ from the one determined during the optimization process.

The presented MIP model assures unique shortest paths if the maximum number of outgoing interfaces T_{max} is set to 1. However, the genetic algorithm does not guarantee that the selected paths are unique shortest paths. Since the metric values are chosen randomly, equal-cost paths are possible. To break equalities and select only one outgoing interface, a certain deterministic procedure is necessary, which allows reproducible results. Therefore, in this work we assume that every interface has a number assigned to it (e.g., IP address) and that routers select their outgoing interfaces based on these numbers. In case of several equal-cost paths and deactivated ECMP feature, a router always chooses the interface with the lowest number. However, it is easily possible to change this selection mode and implement any other decision process as long as it has a predictable behavior.

Nevertheless, to remedy the ambiguity of equal-cost paths in single-path configurations a mixed-integer program was developed, which is applied to an optimized routing configuration (after the HGA has returned the solution). The program exploits the fact that for routing optimization only a small range of metric values is utilized (e.g., using $W_{max} = 20$ instead of 65535). Therefore, it is possible to first scale the metric values so that they use the whole available metric space. Then individual metrics are slightly modified in order to break equalities without changing the shortest paths. In most cases this approach allows solving existing conflicts in order to obtain a configuration with unique shortest paths.

3.6 Routing Adaptation with MPLS

In this section routing adaptation, which utilizes MPLS technology, is considered. Instead of optimizing routing by setting the OSPF or EIGRP metrics, individual flows are now selected and routed explicitly along LSPs. The advantage of this approach is its great flexibility.

The establishment of LSPs is not subject to any restrictions regarding route selection or forwarding mechanisms. All LSPs can be set up independently of each other, in a way to best serve network QoS. The great drawback, however, is that MPLS technology has to be installed in the network. While conventional IGP routing technology is readily available in every IP network, this is not the case for MPLS. MPLS requires special routers at the ingress nodes of the network, which classify all incoming traffic and attach the appropriate labels. Within the network, routers have to be able to switch the packets according to their labels. Finally, at the egress the labels have to be removed again. In addition to the great cost factor, setting up a large number of LSPs requires extra administrational effort as this is mostly done manually.

Consequently, our optimization approach regards MPLS only as a complementary technology to OSPF or EIGRP. Routing optimization is carried out in a hybrid way. Instead of turning every end-to-end flow into an LSP, we would like to keep the cost for the new technology as low as possible. This means that a some flows are selected for label-switching and new equipment is only installed where needed. Most of the flows are still routed by conventional IP technology, following an existing routing scheme.

One objective of this type of routing optimization is again to enhance network QoS by reducing the maximum link utilization in the network. However, since hybrid routing adaptation leads to extra expenses, we also look at the problem from a network planner's perspective. Instead of improving QoS, we would rather like to minimize expenses while meeting a certain desired QoS demand. This means that given a certain maximum utilization the label switched paths are chosen such that the number of required MPLS routers is minimized.

The selection of LSPs and their routing is done by means of mixed-integer programming. Two different models are specified: a flow-based and a path-based approach. The flow-based model considers the whole solution space, while the path-based model works on basis of path sets, restricting the solution space from the start on. In the following paragraphs the models are introduced and their advantages and disadvantages are discussed. Furthermore, we will show how both of them can be used to find good solution.

3.6.1 Mixed-Integer Programming Models

The network is again modeled as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The nodes in $\mathcal{V}$ represent routers, which run a conventional IGP such as OSPF or EIGRP. Without loss of generality, we assume that all nodes could be upgraded to support MPLS, as an ingress, egress, and label-switching router.

3.6.1.1 Common Parameters, Variables and Objective Function

A capacity value C_{ij} is associated with every link (i, j). Furthermore, a number of traffic demands exist, which need to be routed through the network, starting at a flow's ingress node and terminating at the egress node. In contrast to the optimization model of section 3.5.1, the individual flows are now identified by a set $\mathcal{F}$ where each flow f, with $f \in \mathcal{F}$, has an average bit rate of $\hat{A}^f$. The ingress and egress nodes of flow f are denoted as $In(f)$ and $Eg(f)$, respectively. In the following we will refer to end-to-end flows as Ingress/Egress flows or I/E flows in short. On basis of a given routing pattern (e.g., determined by OSPF for a certain link metric vector), the initial route of each I/E flow can be determined. Thus, it is possible to specify the amount of traffic A^f_{ij}, which flow f initially contributes to the total

load on link (i, j) before LSP optimization is performed.

Based on these parameters, the best flows for MPLS routing adaptation need to be identified. However, since we want to keep the number of label-switched flows small, we introduce a limit parameter F_{max}. At this point it is important to note that the number of label-switched I/E flows (LSFs) is not necessarily equal to the number of established LSPs. In case of load sharing it is possible that one flow is split up over several LSPs and that the traffic is arbitrarily distributed across all LSPs. In this case the number of LSPs is larger than the number of LSFs. In order to control not only the number of LSFs, but also to restrict the number of LSPs per LSF we introduce a limit L_{max}. If desired, this limit can be I/E-flow dependent, i.e., there could be a specific L^f_{max} for every flow f. Furthermore, we can also introduce a maximum for the total number of LSPs $\hat{L}_{max}$, irrespective to which I/E flow they belong. Finally, we would like to be able to set a limit U_{max} for the maximum link utilization, which should not be exceeded after the optimization, and a limit R_{max} for the number of MPLS-enabled routers in the network.

The binary variables y^f indicate whether flow f is switched by MPLS ($y^f = 1$) or whether it is routed along the original IGP path ($y^f = 0$). Another set of binary variables, v_i, indicate the routers' technology, either MPLS or regular IP. All nodes i, which are traversed by an LSP, need to support MPLS, i.e., $v_i = 1$ otherwise $v_i = 0$. Finally, variables ρ_{ij} and ρ_{max} are introduced, which hold the utilization of links (i, j) and the network-wide maximum thereof.

As mentioned above, our objective function incorporates two main components: the maximum utilization as well as the total number of required MPLS routers (3.44). The emphasis of the optimization can be varied by setting the weight parameters w_{util} and w_{cost} appropriately. Furthermore, we add a third term, which will lead to the minimization of the total number of label-switched flows. This way it can be assured that only the necessary number of OSPF flows is subject to MPLS, even if the limit F_{max} would allow a higher number. However, as this is not our main objective, weight w_{flow} is usually orders smaller than w_{util} or w_{cost}. Table 3.3 lists the parameters and variables of the mixed-integer programs.

$$z = \min \left(w_{util} \cdot \rho_{max} + w_{cost} \cdot \sum_{i \in \mathcal{V}} v_i + w_{flow} \cdot \sum_{f \in \mathcal{F}} y^f \right) \tag{3.44}$$

3.6.1.2 Flow-based Formulation (F-LSP)

The flow-based model is a mixed-integer extension of the well-known multicommodity flow problem. Variables x^f_{ij} specify the share of flow f, which is carried by MPLS and traverses link (i, j).

On basis of y^f, (3.45) establishes the necessary balance equations for a flow at its source, destination, and intermediate nodes. With (3.46) the maximum number of LSFs is limited to F_{max}. Constraints (3.47) and (3.48) force every node to be marked as an MPLS router whenever an LSP passes it. The maximum number of MPLS routers in the network is R_{max} (3.49). Constraints (3.50) specify the utilization of every link (i, j). The right hand side of the constraints consists of all LSP-carried traffic plus the portion of the traffic still routed by OSPF. For all links, the utilization has to be smaller than the variable ρ_{max} (3.51).

node set	$\mathcal{V}$
edge set	$\mathcal{E}$
flow set	$\mathcal{F}$
set of path alternatives	$\mathcal{P}^f$
set of nodes, edges along a path p	$\mathcal{N}(p)$, $\mathcal{L}(p)$
link capacities	C_{ij}
average bit rate of flow f	$\hat{A}^f$
initial traffic of flow f on link (i,j)	A_{ij}^f
maximum number of LSFs	F_{max}
maximum number of LSPs per LSF	L_{max}
maximum number of LSPs in total	$\hat{L}_{max}$
threshold for link utilization	U_{max}
maximum number of MPLS routers	R_{max}
label-switching indicator variable	y^f
MPLS router indicator variable	v_i
utilization/maximum utilization variable	ρ_{ij}, ρ_{max}
link flow variable	x_{ij}^f
path flow variable	q_p^f
path-in-use indicator	b_p^f

Table 3.3: Parameters and variables of mixed-integer programs for MPLS routing adaptation

$$\sum_{j:(i,j)\in\mathcal{E}} x_{ij}^f - \sum_{j:(j,i)\in\mathcal{E}} x_{ji}^f = \begin{cases} y^f & \forall i \in \mathcal{V}, \forall f \in \mathcal{F} : In(f) = i \\ \text{-}y^f & \forall i \in \mathcal{V}, \forall f \in \mathcal{F} : Eg(f) = i \\ 0 & otherwise \end{cases} \tag{3.45}$$

$$\sum_{f\in\mathcal{F}} y^f \leq F_{max} \tag{3.46}$$

$$v_i \geq x_{ij}^f \qquad \forall (i,j) \in \mathcal{E}, \forall f \in \mathcal{F} \tag{3.47}$$

$$v_j \geq x_{ij}^f \qquad \forall (i,j) \in \mathcal{E}, \forall f \in \mathcal{F} \tag{3.48}$$

$$\sum_{i\in\mathcal{V}} v_i \leq R_{max} \tag{3.49}$$

$$\rho_{ij} \cdot C_{ij} = \sum_{f \in \mathcal{F}} x_{ij}^f \cdot \hat{A}^f + \sum_{f \in \mathcal{F}} \left(1 - y^f\right) \cdot A_{ij}^f \qquad \forall (i,j) \in \mathcal{E} \tag{3.50}$$

$$\rho_{ij} \leq \rho_{max} \qquad \forall (i,j) \in \mathcal{E} \tag{3.51}$$

$$\rho_{max} \leq U_{max} \tag{3.52}$$

$$0 \leq x_{ij}^f \leq 1 \qquad \forall (i,j) \in \mathcal{E}, \forall f \in \mathcal{F} \tag{3.53}$$

$$y^f \in \{0,1\} \qquad \forall f \in \mathcal{F} \tag{3.54}$$

Solving the given optimization problem leaves us with possibly several LSPs per LSF since x_{ij}^f can be fractional. By allowing only binary x_{ij}^f we can restrict the number of LSPs per flow to one. However, this drastically increases the complexity of the model. Another shortcoming would still be that the flow-based model does not provide an effective way to limit the number of LSPs per LSF. Therefore, the following path-based model is specified, which can either be used by itself or in combination with the flow-based model.

3.6.1.3 Path-based Formulation (P-LSP)

In order to reduce the complexity of the optimization model, a set of path alternatives $\mathcal{P}^f = \{P_1^f, P_2^f, ...\}$ is specified for every I/E flow f. Each element p of $\mathcal{P}^f$ describes a potential LSP, starting at node $In(f)$ and terminating at $Eg(f)$. The nodes and edges along such a path $p \in \mathcal{P}^f$ are contained in the sets $\mathcal{N}(p)$ and $\mathcal{L}(p)$, respectively.

Variables q_p^f correspond to the ratio of traffic flow f that is transmitted over label-switched path p. In order to indicate whether an LSP alternative p is indeed being used for flow f, binary variables b_p^f are introduced.

The following constraints describe the optimization problem. They are mostly analog to the ones of the flow-based model above. Whenever flow f is label-switched ($y^f = 1$), equations (3.55) specify that the total load has to be distributed across all possible LSPs. With (3.56), the number of LSFs is kept below a desired maximum, and constraints (3.57) make sure that nodes, which are located along LSPs, are marked as MPLS routers. Again, the number of MPLS routers cannont exceed a certain threshold (3.58). Constraints (3.59) and (3.60) take care of the definition of link utilization values.

However, in contrary to the flow-based model we are now able to indicate utilized LSPs (3.62) and, thus, set a threshold for the maximum number of LSPs per I/E flow (3.63) as well as for total number of LSPs (3.64).

$$\sum_{p \in \mathcal{P}^f} q_p^f = y^f \qquad \forall f \in \mathcal{F} \tag{3.55}$$

$$\sum_{f \in \mathcal{F}} y^f \leq F_{max} \tag{3.56}$$

$$q_p^f \leq v_i \qquad \forall f \in \mathcal{F}, \forall p \in \mathcal{P}^f, \forall i \in \mathcal{N}(p) \tag{3.57}$$

$$\sum_{i \in \mathcal{V}} v_i \leq R_{max} \tag{3.58}$$

$$\rho_{ij} \cdot C_{ij} = \sum_{f \in \mathcal{F}} \sum_{p \in \mathcal{P}^f : \mathcal{L}(p) \ni (i,j)} q_p^f \cdot \hat{A}^f + \sum_{f \in \mathcal{F}} \left(1 - y^f\right) \cdot A_{ij}^f \qquad \forall (i,j) \in \mathcal{E} \tag{3.59}$$

$$\rho_{ij} \leq \rho_{max} \qquad \forall (i,j) \in \mathcal{E} \tag{3.60}$$

$$\rho_{max} \leq U_{max} \tag{3.61}$$

$$q_p^f \leq b_p^f \qquad \forall f \in \mathcal{F}, \forall p \in \mathcal{P}^f \tag{3.62}$$

$$\sum_{p \in \mathcal{P}^f} b_p^f \leq L_{max} \qquad \forall f \in \mathcal{F} \tag{3.63}$$

$$\sum_{f \in \mathcal{F}} \sum_{p \in \mathcal{P}^f} b_p^f \leq \hat{L}_{max} \tag{3.64}$$

$$0 \leq q_p^f \leq 1 \qquad \forall f \in \mathcal{F}, \forall p \in \mathcal{P}^f \tag{3.65}$$

$$y^f \in \{0,1\} \qquad \forall f \in \mathcal{F} \tag{3.66}$$

$$b_p^f \in \{0,1\} \qquad \forall f \in \mathcal{F}, \forall p \in \mathcal{P}^f \tag{3.67}$$

$$v_i \in \{0,1\} \qquad \forall i \in \mathcal{V} \tag{3.68}$$

3.6.1.4 Multi-Service Considerations

Although not explicitly considered in the above formulations, the hybrid routing adaptation approach is very well suited for multi-service environments. Analog to the multi-service extension of the OSPF/EIGRP routing optimization model in section 3.5.1.6, service-specific flow and utilization variables as well as bandwidth share parameters could also be introduced in the F-LSP and P-LSP models. Furthermore, similar considerations apply for possible objective functions.

3.6.2 Optimization Procedures

With the two optimization models at hand, MPLS-based routing adaptation can be approached from several directions.

First of all, *F-LSP* can be solved. If the outcome is acceptable, it represents the best possible solution. However, if the number of LSPs per flow is too large, additional computations are necessary. Furthermore, the minimization of the maximum utilization and/or the number of MPLS routers might produce LSPs, which are too long (e.g., in terms of hop count).

Therefore, solving *P-LSP* might be necessary. As mentioned above, the lower complexity allows controlling the number of LSPs explicitly. Furthermore, by defining the sets of path

alternatives before carrying out the optimization, it can be assured that the result will contain only valid routes.

In this work, we use three different path-set generators:

- k edge-disjoint shortest paths: To find the set of k edge-disjoint shortest paths Bhandari's algorithm is applied [Bha99]. The advantage of this approach is that the selected LSPs are taking different routes through the network. Therefore, this approach could be chosen if resilience is desired.

- All paths up to a certain length: This method generates larger sets of paths. The paths are computed by performing a breadth-first search for every flow.

- Paths derived from *F-LSP* flow variables: In case the flow-based model returns a solution with only a few more LSPs than desired, a consecutive *P-LSP* optimization can be used to slightly modify the result and make it valid. The algorithm that generates paths from flow variables is described below.

Deriving Label-Switched Paths from Flow Variables

The *F-LSP* model returns the values for all flow variables x_{ij}^f. Initially, these values do not have a notion of paths associated with them. However, in order to interpret the routing solutions and to check their validity, we need to form paths, which are consistent with the values of the flow variables. This means that for LSFs, i.e., flows f with $y^f = 1$, we have to set up possibly several LSPs, specify their routes through the network, and assign the amount of carried traffic to each of them. It is easily understood that there exists an infinite number of possibilities to do so. Just take an example with exactly one MPLS path through the network. It is possible to define any number of parallel LSPs, which carry an arbitrary amount of load, as long as the sum of the individual load shares equals 1. However, considering our context of routing adaptation, the number of LSPs should be kept low. Therefore, we propose following procedure to find the set of LSPs for a certain I/E flow:

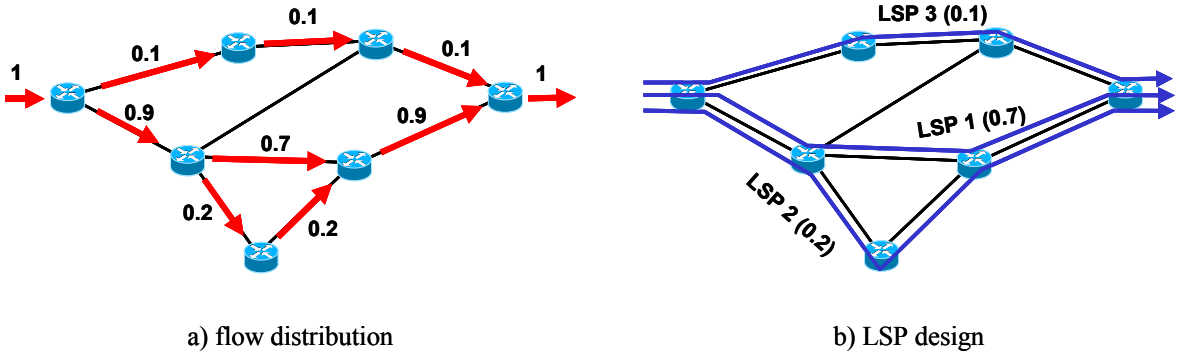
1. Find a new LSP:
   Starting at the ingress node, iteratively traverse the outgoing link with the largest flow value until the egress node is reached. Remove possible loops.

2. Assign the load share to the new LSP:
   The LSP's load share is equal to the minimum of all flow values along the new path.

3. Subtract the LSP's load share from all flow values along the LSP's path.

4. If no more LSPs with load share larger than 0 can be found, stop. Otherwise, go to step 1.

The concept of the algorithm is illustrated in Figure 3.12. The LSPs are labeled in the order of their generation.



a) flow distribution b) LSP design

Figure 3.12: Deriving LSP design from flow distribution

3.7 Numerical Results and Discussion

In this section we present typical results of routing optimization and adaptation. The applicability of the algorithms and optimization programs is demonstrated by means of various sample topologies. If not mentioned otherwise, the networks considered in this chapter are the ones described in Appendix A. The number of nodes, edges, and flows of these networks are summarized in Table 3.4. All links have the same capacity and traffic flows were generated randomly. The following computations were all performed on PCs with Pentium 4 processors (at least 1.4 GHz), running the Linux operating system.

	N11	**N20**	**N40**	**N50**	**N100**
# nodes $\|\mathcal{V}\|$	11	20	40	50	100
# links $\|\mathcal{E}\|$	48	102	150	126	330
# flows $\|\mathcal{F}\|$	110	380	500	2450	9900

Table 3.4: Network scenarios

3.7.1 Routing Optimization with OSPF and EIGRP

We first investigate routing optimization and adaptation based on the interior gateway protocols OSPF and EIGRP.

3.7.1.1 Applicability of the MIP Model

The MIP model can be validated for small networks. However, the network scenarios considered for the mixed-integer program are different from the ones given in the appendix as these are too large to be solved. At this point we are not interested in the actual results of the optimization program, i.e., the final maximum link utilization value in the network. We rather would like to demonstrate the applicability of the models and point out the difference in complexity of the two protocols. Table 3.5 summarizes the complexity of the optimization problems (number of rows and columns of the optimization system, as well as the number of non-zero elements in the matrix) for the sample scenarios. Furthermore, the solution times are given in seconds. As we can see, the OSPF model, which is a reduced form of the model

given in section 3.5.1, can still be solved optimally within short amount of times (in some scenarios $t < 1$ second), while the complexity and the solution times of the EIGRP model increase quickly. For the network with 8 nodes and 22 links, the solver does not return an EIGRP solution.

			OSPF				EIGRP			
$\|\mathcal{V}\|$	$\|\mathcal{E}\|$	$\|\mathcal{F}\|$	rows	cols	non-zeros	t	rows	cols	non-zeros	t
5	12	20	306	114	1146	<1 s	1406	644	4486	7 s
6	18	30	600	187	2345	< 1 s	2017	824	6764	116 s
7	20	42	923	275	3700	193 s	2853	1149	9650	213 s
8	22	32	1126	363	4480	285 s	3601	1509	11968	inf

Table 3.5: Complexity and solution times for the MIP approach

Conclusion

It can be shown that the MIP models work correctly and can be applied to small networks. While the reduced OSPF model can still be used for somewhat larger networks, the applicability of the comprehensive EIGRP model is limited. For larger networks, we can use the model only for a subset of the nodes (e.g., the core network) or we have to deploy the presented heuristic.

3.7.1.2 Comparison of OSPF and EIGRP Optimization

The hybrid genetic algorithm for routing optimization has been applied to our sample network scenarios. At first, the focus is on global routing optimization without consideration of the preexisting weight settings, which had a weight of 10 assigned to each link (for *delay* as well as *icm*). Since all *icm* values are equal, OSPF and EIGRP lead both to the same routing. The initial maximum utilization values for single-path and equal-cost multi-path (ECMP) configurations and the optimization results are summarized in Table 3.6. It shows for every scenario the optimized maximum utilization value ρ_{max}. Furthermore, the number of modified link weights n_{metric} and the number of rerouted (affected) flows $n_{rerouted}$ that are necessary to achieve the best solutions are given. Additionally, the computation times t are listed. Since genetic algorithms do not guarantee to return the optimum, the computations were carried out several times for each scenario with various parameter settings. The table contains the best solution for each scenario and the computation times correspond to the run, which returned this best solution.

The most important observation is that in all cases the maximum link utilization can be drastically reduced through routing optimization, often achieving a utilization that comes close to the lower bound. However, it is usually required to modify a rather high portion of link weights. In some cases, over 90% of the link metrics have to be changed, affecting up to 60% of all flows (e.g., N20, OSPF ECMP). As expected, EIGRP is able to achieve lower utilization values than OSPF, and load sharing (although it is quite limited for destination-based routing protocols) often allows further QoS enhancement. However, depending on the topology and the load situation, the advantage of EIGRP optimization over OSPF varies.

Figure 3.13 demonstrates the benefits of routing optimization based on OSPF and EIGRP. In Figure 3.13a the utilization is given as a factor of the lower-bound (i.e., all utiliza-

		N11	N20	N40	N50	N100
Standard Hop-by-Hop Routing						
Single Path	ρ_{max}	1.0446	1.0966	0.8692	1.0446	1.3087
ECMP	ρ_{max}	0.7608	0.7911	0.8819	0.7608	1.1626
Optimized Routing						
Lower Bound	ρ_{max}	0.5476	0.4364	0.2596	0.5476	0.57
OSPF	ρ_{max}	0.6079	0.4604	0.4240	0.6079	0.6730
	n_{metric}	43 (90%)	95 (93%)	101 (67%)	43 (34%)	290 (88%)
	$n_{rerouted}$	32 (29%)	159 (42%)	113 (23%)	657 (27%)	5631 (57%)
	t	225 s	259 s	632 s	225 s	29.7 min
OSPF ECMP	ρ_{max}	0.5702	0.4583	0.4060	0.5702	**0.6405**
	n_{metric}	38 (79%)	92 (90%)	78 (52%)	38 (30%)	311 (94%)
	$n_{rerouted}$	33 (30%)	232 (61%)	217 (43%)	695 (28%)	7096 (72%)
	t	12.7 min	10.5 min	574 s	12.7 min	3.6 h
EIGRP	ρ_{max}	0.5934	0.4476	**0.3617**	0.5576	0.6582
	n_{metric}	92 (96%)	194 (95%)	282 (94%)	214 (85%)	631 (96%)
	$n_{rerouted}$	33 (30%)	170 (45%)	150 (30%)	550 (22%)	5963 (60%)
	t	368 s	334 s	315 s	169 s	1.2 h
EIGRP ECMP	ρ_{max}	**0.5678**	**0.4406**	**0.3617**	**0.5574**	**0.6405**
	n_{metric}	92 (96%)	193 (95%)	282 (94%)	150 (60%)	311(47%)
	$n_{rerouted}$	32 (29%)	220 (58%)	203 (41%)	657 (27%)	7096 (72%)
	t	180 s	424 s	315 s	19.8 min	3.6 h

Table 3.6: Results of OSPF/EIGRP routing optimization

tion values are normalized respective to the lower-bound). Higher values correspond to lower network QoS. A value of 1 indicates that the lower bound, i.e., the best possible utilization is achieved. To illustrate the reduction, the maximum utilization of the original configuration with hop-based single-path routing is shown. For the sake of clarity, the value for hop-based ECMP routing is omitted. In order to show the higher optimization potential of EIGRP, the graph in Figure 3.13b depicts the improvement of network QoS (the additional utilization reduction), which can be achieved by EIGRP optimization over OSPF optimization. The differences between the two approaches are given as fractions of the OSPF results, i.e., $(\rho_{max}(OSPF) - \rho_{max}(EIGRP))/\rho_{max}(OSPF)$ and $(\rho_{max}(OSPF\ ECMP) - \rho_{max}(EIGRP\ ECMP))/\rho_{max}(OSPF\ ECMP)$. In some scenarios the difference is very small (below 2% in *N50*, ECMP and *N100*, Single Path) or even zero (*N11*, ECMP and *N100* ECMP). However, for some topologies the improvement can be striking (16% in *N40*, Single Path). This observation is consistent with the discussion of routing optimization principles earlier in this chapter. EIGRP optimization performs especially well in networks with hierarchical structures such as *N40* and *N50*. In these scenarios it can play out its advantage of being able to differentiate routes with common intermediate nodes by means of the *icm* link weights.

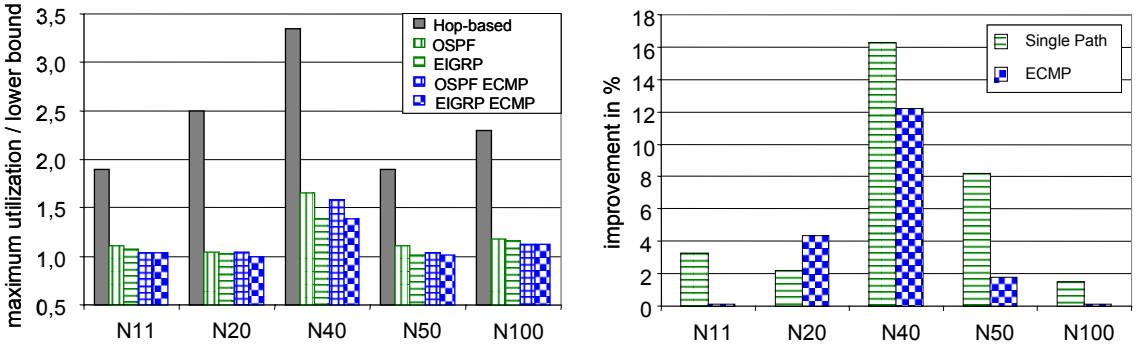


Figure 3.13: Comparison of optimization results for OSPF and EIGRP

Conclusion

Traffic engineering based on interior routing protocols together with optimized weight setting is an appropriate means to increase QoS in a network with unevenly distributed traffic. Furthermore, considering not only an additive metric as it is used in OSPF but also a concave one such as with EIGRP can often improve routing optimization considerably.

3.7.1.3 Routing Adaptation with OSPF and EIGRP

In this section, we investigate routing optimization where only a certain number of metric values are modified. Routing adaptation, which focuses on the number of rerouted flows instead of modified metrics, behaves similarly. Therefore, the discussion of the outcomes is omitted here. We also leave out EIGRP-based results as they were very similar to OSPF-based ones. For routing optimization with a limited number of metric changes, EIGRP did not

show any clear advantage over OSPF. If the limit of the metric changes was set large enough, EIGRP could certainly outperform OSPF in some scenarios, however, this corresponds more to global routing optimization as discussed in the preceding section.

Routing adaptation was carried out by means of our hybrid genetic algorithm. The adaptation feature is activated by setting weight w_{metric} (see objective function (3.41)) to a positive value. Furthermore, parameter M_{max} fixes the threshold for the number of admissible metric changes. In all following experiments, w_{flow} was set to zero, i.e., the number of modified flows was not taken into account explicitly.

Figure 3.14 illustrates the tradeoff between network QoS and the number of metric changes for scenario *N11* with OSPF and OSPF ECMP. As expected, network QoS improves with increasing number of metric changes. Furthermore, a nice observation is that the greatest gains can be achieved with just a few metric changes. In case of OSPF, changing only 5 link metrics reduces the maximum utilization from 1.04 to 0.70. In order to approach the best-known solution (see Table 3.6) a lot more metric modifications are necessary. Therefore, if every single metric modification causes a short interruption of the network service due to transition phases in the routing process, an operator might be willing to accept this tradeoff and give up some of the utilization reduction.

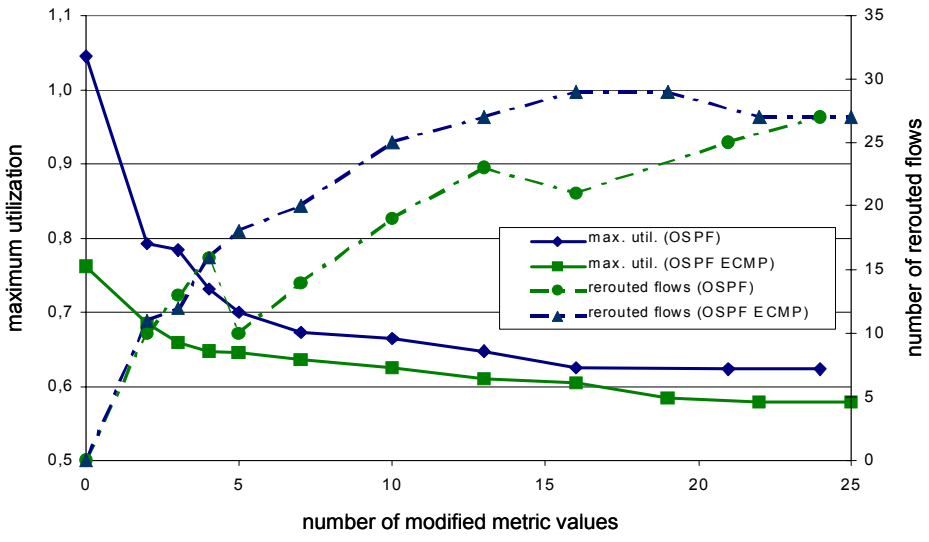


Figure 3.14: Tradeoff between network QoS and number of metric changes

Table 3.7 summarizes results for optimized routing adaptation based on OSPF with and without ECMP. For each scenario we have set the limit of metric changes M_{max} to around 10% of the number of links. The table lists the final maximum utilization values ρ_{max}, the necessary number of rerouted flows $n_{rerouted}$, as well as the obtained routing adaptation gain. The value of *gain* is the QoS improvement in relation to the adaptation potential. The potential is the maximum possible improvement, which can be expected from the adaptation process. Thus, $gain = (\rho^{init}_{max} - \rho_{max})/(\rho^{init}_{max} - \rho_{best})$ where ρ^{init}_{max} is the maximum utilization of the original configuration and ρ_{best} is the best solution according to Table 3.6.

In all networks the maximum link utilization ρ_{max} can be reduced considerably by only changing 10% of the link weights. This is especially noticeable for OSPF routing where the

gain is always above 69%. For *N40* the gain is even 95%, indicating that with only 15 metric changes the best solution is almost achieved. Note that this number is a lot smaller than the number of metric changes given in Table 3.6 for scenario *N40* with OSPF. However, at that point the algorithm did not pay any attention to the number of modified metrics at all.

For OSPF ECMP adaptation, the QoS improvement is also significant as compared to the original hop-based configuration. However, since the traffic is already more evenly distributed from the beginning on, great enhancement steps as with OSPF are not possible. Thus, the gain values are not quite as high. Anyway, the absolute utilization values are usually about equal to or better than the OSPF results.

	N11	**N20**	**N40**	**N50**	**N100**
n_{metric}	5	10	15	13	33
OSPF					
ρ_{max}	**0.70**	**0.66**	**0.45**	**0.67**	**0.87**
$n_{rerouted}$	10	62	75	306	2453
$gain$	79%	69%	95%	86%	69%
OSPF ECMP					
ρ_{max}	**0.65**	**0.56**	**0.47**	**0.65**	**0.88**
$n_{rerouted}$	17	89	112	440	3691
$gain$	58%	70%	87%	58%	54%

Table 3.7: Results for OSPF routing adaptation

Conclusion

The results show that for routing adaptation it is not necessary to change a large number of metrics. Even a few metric modifications can already suffice to bring down the maximum utilization and, thus, increase network QoS drastically. Comparing OSPF with EIGRP, our computations have not shown any advantages of EIGRP over OSPF, if only a limited number of metric changes is allowed.

3.7.2 Routing Adaptation with MPLS

In this section MPLS-based routing adaptation is investigated. The F-LSP and P-LSP models are applied to our sample scenarios, which are initially loaded with the given traffic flows. It is assumed that OSPF with single-path configuration is used as the underlying conventional routing protocol. Based on these load scenarios LSPs are introduced in order to improve QoS.

At first, we focus on the improvement of QoS without taking into account the number of required LSRs (QoS-driven approach). Then, in the second part the tradeoff between QoS improvement and the number of LSRs is considered (cost-driven approach).

3.7.2.1 QoS-driven Approach: Emphasis on Utilization

This approach puts sole emphasis on the minimization of the maximum link utilization. This is achieved by setting the weight parameters of the objective function (3.44) appropriately, i.e., $w_{util} = 1$ and $w_{cost} = 0$. Furthermore, in order to reduce the complexity of the optimization

		N11	N20	N40	N50	N100
Initial Load	ρ_{max}	1.04	1.10	0.87	1.04	1.31
LB	ρ_{max}	0.55	0.44	0.26	0.55	0.57
MPLS Routing Adaptation						
5% I/E flows	ρ_{max}	**0.66**	**0.47**	**0.52**	**0.55**	**0.92**
	I/E-flows	6	19	25	123	493
	LSPs	8	28	26	142	496
	t	2 s	1356 s	4 s	153 s	3.3 h
10% I/E flows	ρ_{max}	**0.56**	**0.44**	**0.37**	–	**0.83**
	I/E-flows	11	38	50	–	990
	LSPs	31	62	61	–	998
	t	4 s	15 s	9 s	–	2.9 h
20% I/E flows	ρ_{max}	**0.55**	–	**0.26**	–	**0.78**
	I/E-flows	22	–	100	–	1980
	LSPs	43	–	171	–	1830
	t	4 s	–	906 s	–	2.5 h

Table 3.8: Results of MPLS routing adaptation

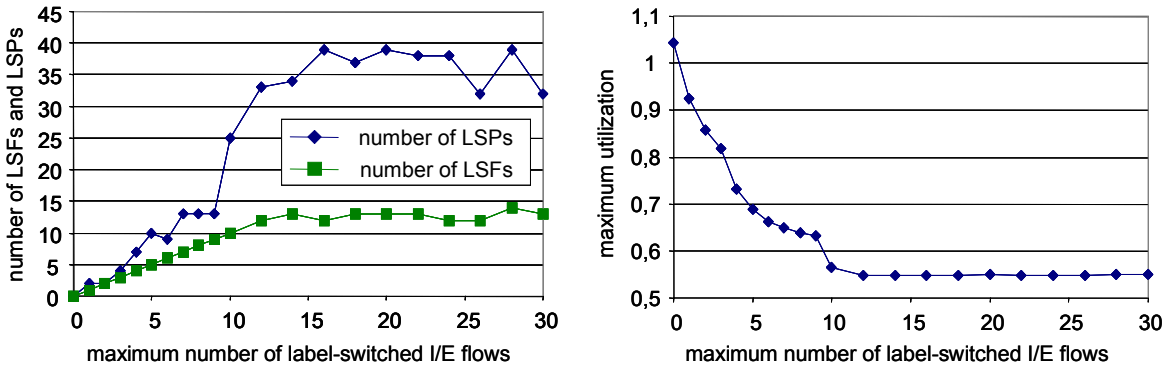
models, it is possible to deactivate the constraints that deal with variable v_i. These are irrelevant whenever $w_{cost} = 0$.

Table 3.8 summarizes important results of MPLS routing adaptation for different maximum numbers of label-switched flows (I/E flows). The limit was set to 5%, 10%, and 20% of all flows. In case of scenarios *N11*, *N20*, *N40*, and *N50* the F-LSP model was applied. However, scenario *N100* is too complex to be solved with the flow-based model. Therefore, the P-LSP model was used for this network where the sets of admissible paths between all nodes were computed by the 2-edge-disjoint shortest path algorithm. To achieve best results, we at first do not restrict the number of LSPs per label-switched I/E flow (MPLS multi-path). The number of LSPs, which were returned by the solver, are listed in the table. Furthermore, the computation times are recorded. The programs were solved with an optimality gap of up to 4%.

It can be observed that with 5% label-switched I/E flows the maximum utilization can already be greatly reduced. In case of scenario *N50*, the lower bound is even achieved and further increase of the number of label-switched I/E flows is not necessary. For *N20*, the optimum is reached with 10% label-switched flows, for *N11* and *N40*, 20% are enough. As the size of *N100* requires the use of P-LSP with only two path options per I/E flow, the optimum is not achieved in our computations. Nevertheless, the improvement is considerable.

To demonstrate the characteristics of MPLS routing adaptation in more detail, consider Figure 3.15. For network scenario *N11* the maximum number of label-switched I/E flows F_{max} is increased from 0 to 30 (the total number of flows is 110) and the F-LSP model is solved. Figure 3.15a shows the final number of label-switched I/E flows (LSFs) and the resulting number of LSPs. The latter is always equal to or larger than the former since every LSF requires at least one LSP. The most striking observation is that from 13 label-switched I/E flows on, the number of flows, which the solver selects for label-switching, stays the same (due to 1% optimization gap it varies between 13 and 14). The reason for this is that the

minimum of the maximum utilization 0.55 is already reached (see Figure 3.15b). No further OSPF flows need to be subject to MPLS. The number of LSPs at this point varies between 35 and 40.



a) Number of LSFs and LSPs

b) Maximum utilization

Figure 3.15: MPLS routing adaptation for scenario *N11*

So far, it was possible that one label-switched flow was split up over several LSPs. In Figure 3.16, different strategies are compared for *N11*, that return exactly one LSP per LSF. Method 1 (*F-LSP + P-LSP*) applies the F-LSP model first and solves a follow-up P-LSP problem, which is based on the paths derived from F-LSP. The remaining methods use only the P-LSP model. They differ in the sets of admissible paths. For the *2-EDSP* and *3-EDSP* methods, the path sets were computed by the 2- and 3-edge-disjoint shortest-path algorithms, respectively. In case of *all-2-factor* all paths are included, which are up to twice the hop-length as the original shortest path. The paths were computed using a depth-first-search algorithm. Furthermore, the lower bound is illustrated.

We can observe that for a small number of LSPs (up to 5) the various methods work equally well. However, for larger numbers of LSPs, the *F-LSP + P-LSP* method shows some deficiencies. The reason for this is that the F-LSP model does not leave many path options for the consecutive P-LSP to choose from. Taking for example the result of F-LSP for 12 LSFs, we have only a total of 34 LSPs, which serve as path options for exactly 12 I/E flows (see Figure 3.15). Thus, the solution space is already very restricted for the following P-LSP program to start with. However, once the LSP constellations are computed by F-LSP, the effort to find the corresponding P-LSP solution is negligible. As the obtained utilization values might suffice in some cases, it is worthwhile computing them. If they are not good enough, they at least provide an upper bound and give an indication of what can be achieved by the P-LSP model with a higher number of path alternatives.

Conclusion

It could be shown that MPLS adaptation is a good means for traffic engineering in order to improve network QoS. For all procedures, we receive very good results. However, it is also noticeable that a greatly restricted path set limits the potential of the routing adaptation algorithm.

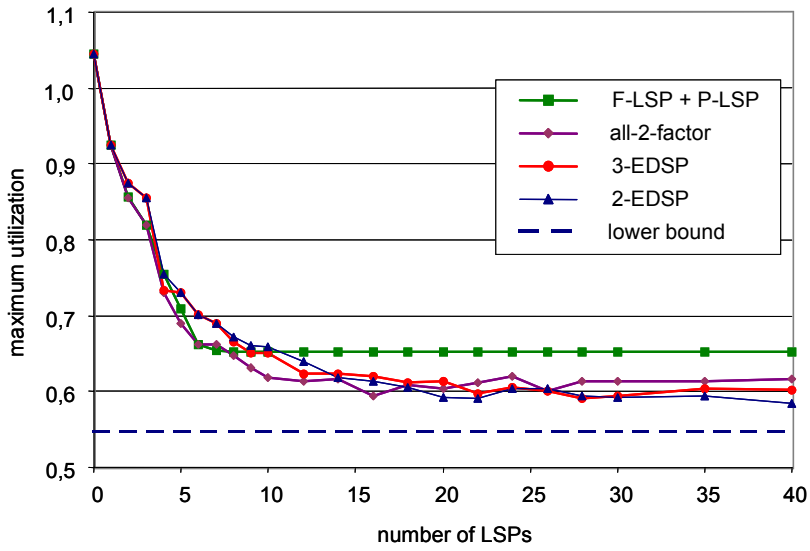


Figure 3.16: Comparison of single-LSP strategies

3.7.2.2 Cost-driven Approach: Tradeoff between QoS and Number of LSRs

After focusing only on QoS improvement, we now shift our attention to the number of necessary MPLS routers. Different strategies exist to do so.

By setting weight parameter $w_{cost} = 1$ and $w_{util} = 0$, the number of MPLS routers is minimized. However, in order to obtain meaningful results, the limit for the maximum utilization U_{max} has to be adjusted appropriately. Starting from a higher value, U_{max} can be decreased by little intervals until the problem becomes infeasible. For each U_{max} value, one of the models is solved and a solution is returned, which keeps the maximum utilization below the desired threshold and which requires the smallest number of MPLS-enabled routers to do so. The intuition behind this approach is that a desired network QoS should be achieved with the cheapest possible solution.

A second method is to still have $w_{cost} = 0$ and $w_{util} = 1$ (just as in the QoS-driven case). However, now the v_i related constraints are not removed, and a limit for the number of MPLS routers R_{max} is set. If R_{max} equals the number of nodes in the network, the result would be the same as in the QoS-driven scenarios. However, if this threshold is decreased, the optimization process will return solutions, which minimize the utilization as long as the number of MPLS routers does not exceed the desired maximum. Here the basic idea is that for a given amount of money (i.e., for a certain number of MPLS routers), the best possible network QoS should be obtained.

The third approach uses the weighted objective function where both weights w_{cost} and w_{util} are positive. The exact behavior of the algorithm then depends on the ratio of the two weights. However, the adjustment of the values is not a straightforward and intuitive process. Effects of certain settings are difficult to predict. Therefore, it is advisable to use one of the first two methods.

Although the three approaches are intuitively different, their final results show the same characteristics. Therefore, we only use here the second method to show the tradeoff between network QoS and the number of MPLS routers. The graph in Figure 3.17 was produced by

the F-LSP model for *N11*, with a limit of 10 for label-switched flows. The graph shows how the number of MPLS routers influences the achievable network QoS. If one is willing to install three MPLS routers (note that two are the minimum if any flow should be label-switched), the maximum utilization can be reduced from 1.04 to around 0.88. For increasing numbers of MPLS routers, the maximum utilization can further be decreased, until the optimum is reached when all routers implement MPLS capability. The graph also shows the resulting numbers of LSPs. It is interesting to see that while the utilization can be decreased by increasing the number of MPLS routers from 3 to 5, the number of LSPs does not have to go up. The QoS improvement is achieved not by selecting more LSPs but by selecting more appropriate ones.

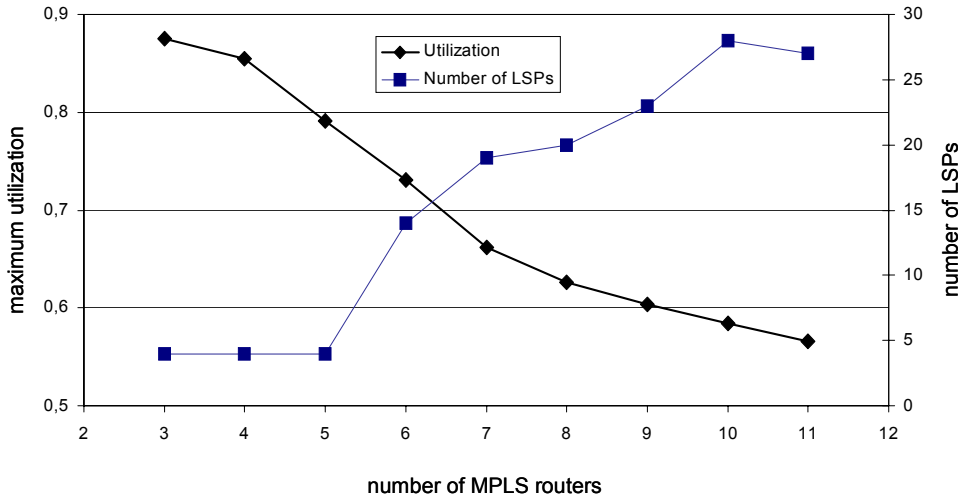


Figure 3.17: Cost-driven MPLS adaptation in network *N11*

Conclusion

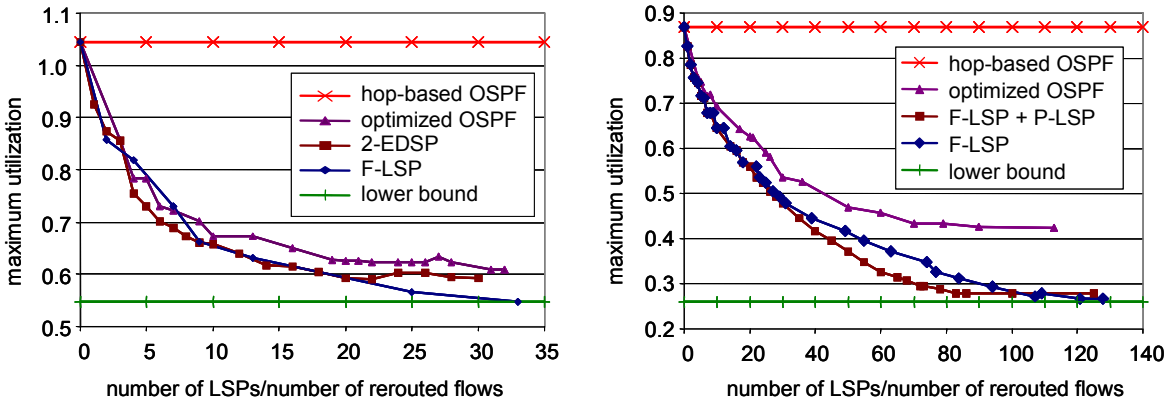
Used in the cost-driven mode, the optimization models provide a handy tool for the support of the migration process from a purely IGP-routed network to an MPLS-enabled network. Instead of replacing every native IP router by an MPLS device, it is conceivable to introduce them only at locations where really required. The algorithms provide a good means to determine the best locations together with the optimum selection of LSPs.

3.7.3 Comparison of IGP and MPLS Routing Adaptation

So far, we have shown how routing adaptation can be performed with conventional routing protocols on the one side and with MPSL on the other. We now conclude the discussion of the numerical results by comparing the two general approaches with each other. For a network provider who is running a conventional routing protocol it is important to know what can be gained by introducing MPLS as a traffic engineering mechanism as opposed to optimizing the configuration of the existing protocol.

In Figure 3.18 the performance of routing adaptation based on OSPF is compared with MPLS for networks *N11* and *N40*. In order to make the corresponding results comparable,

the number of rerouted flows is given for OSPF (instead of the number of modified link metrics, as done earlier). In both scenarios, OSPF with single-path configuration is applied. The LSPs for scenario *N11* were computed by solving the F-LSP model (*F-LSP* in Figure 3.18a) and by applying the P-LSP model to the path sets computed by the 2-edge-disjoint shortest path algorithm (*2-EDSP*). Note that for *F-LSP* several LSPs per LSF are allowed, while in the *2-EDSP* case no load-sharing was accepted. In case of *N40*, the LSPs were computed by solving the flow-based optimization model (*F-LSP*, with load-sharing) and by a combination of the flow-based model with a consecutive path-based optimization (*F-LSP + P-LSP*, without load-sharing). In both graphs, the lower bound as well as the initial load distribution resulting from hop-based OSPF are indicated.



a) Network scenario *N11* b) Network scenario *N40*

Figure 3.18: Comparison of routing adaptation with MPLS and OSPF

For scenario *N11*, the differences between the two adaptation strategies are rather small. At its best, MPLS can only reduce the maximum utilization by an additional 10% as compared to OSPF optimization. This is if multiple LSPs per LSF are allowed. If we compare the *2-EDSP* approach with optimized OSPF (two single-path configurations), the advantage of MPLS is even smaller. Thus, routing adaptation in this network scenario works already very well with OSPF. MPLS does not have a chance to play out its increased flexibility.

This is different for scenario *N40*, where the global OSPF optimization solution did not come as close to the lower bound as in scenario *N11* (see Table 3.6 above). In the graph of Figure 3.18b we see that with MPLS the maximum link utilization can be reduced to a value below 0.3 if about 100 LSPs are established. OPSF on the other side does not make it below 0.4, which corresponds to a gap of over 30%. Therefore, if a provider would like to keep the utilization in network *N40* below 40% or lower, MPLS would be required as a traffic engineering technology. In case a utilization of 50% is sufficient, optimized OSPF would still suffice and the need for MPLS or capacity extension could be delayed.

3.8 Summary

In this chapter, tools for routing optimization and adaptation based on conventional routing protocols as well as on MPLS were presented. It has been shown how OSPF and EIGRP

can be used for traffic engineering and how MPLS can complement the optimization process. The major goal of all optimization algorithms was the enhancement of network QoS while the traffic load and the capacities in the networks were assumed to be fixed. This way routing optimization can be used as part of the traffic engineering process within network management or as a step of the overall network design process.

Chapter 4

Dimensioning for Elastic Traffic

After the discussion of routing optimization, we bring our attention to network dimensioning, another fundamental step of the overall network design process. In this chapter, a methodology is presented, which allows the dimensioning of networks for elastic traffic. This type of traffic is generated by applications with non-realtime characteristics and is typically carried by the Transmission Control Protocol (TCP). At this point the dimensioning procedure is considered in isolation – independently from any other planning issue.

Every dimensioning procedure requires a model, which provides a relationship between the amount of offered load, the link bandwidth, and the achieved QoS, taking into account the characteristics of the workload and the transmission process. In case of elastic traffic the packet transmission process is mainly affected by the rate-sharing features of TCP. Our network dimensioning procedure is based on the processor sharing model. We extend the model in order to take into account specifics of the TCP protocol and show how it can be applied to dimension not only single links but also networks. The validity of the approach is demonstrated by means of extensive simulations.

4.1 Overview and Objectives

During the dimensioning process, the link capacities or, in case of multiple services, the bandwidth shares, which need too be assigned to elastic traffic, have to be determined. We assume that elastic traffic is not subject to admission control. Thus, no flows are blocked and all incoming traffic is accepted into the network. As we consider the dimensioning process in isolation of any other network planning step, it is further assumed that routing is fixed and that the paths between all nodes can be determined unambiguously. Given a certain demand structure, we are therefore able to derive the amount of elastic traffic on every link. From this, the required bandwidth values can be computed.

Overall, the objective is to minimize the bandwidth costs (CAPEX), while still meeting a desired degree of QoS. The relevant QoS measure for elastic traffic flows is *throughput*, which specifies the amount of data that can be transferred in a certain time period. However, as we are looking at stochastic systems (in practice as well as in theory), it is not realistic to request that all transactions obtain exactly the same throughput. Therefore, we will have to settle for average values taken over many transfers. Specifically, in this work we will consider following two slightly different notions of throughput constraints:

- For a specific file size x_t the average transaction time (over all transfers of this size)

should not exceed a certain threshold T_t. The respective throughput is calculated as x_t/T_t. The motivation behind this interpretation of throughput is the following: Time T_t should reflect a typical customer's waiting-time tolerance level, which, of course, depends on the deployed access technology and the type of applications. If the value for x_t is picked in a way that only a small percentage of all transactions is longer than this value (e.g., the 95th percentile of an assumed file size distribution), the customers will experience a satisfactory QoS most of the times. We argue that a user does not care whether files of smaller size are transmitted proportionally faster or not, as long as the tolerance level is not exceeded.

- For any file size x a minimum average throughput D_t should be achieved, i.e., the average transaction time $T(x)$ for files of size x should be below x/D_t. This constraint is more stringent than the first one since it requires that for all file sizes the average throughput is the same. However, as it will be shown later, there is a bias against small elastic flows, which leads to lower throughput values. Therefore, it might not be possible to dimension a network so that all file sizes achieve the same average throughput.

It is emphasized at this point that we are always dealing with an individual network domain, which could be part of a greater network or even the world-wide Internet. We cannot control the bandwidth assignment and the QoS assurance in networks other than our own. Therefore, the QoS guarantees are only given for the considered network. Whenever throughput degradation for example occurs in the neighboring network, it cannot be made up for by overdimensioning our network domain. It can only be the objective to dimension the network appropriately for the case where other network providers act accordingly.

In the following sections we give an overview of the transmission control protocol, which is the major data transmission protocol in today's Internet. As this protocol greatly influences the data transfer process and, thus, the dimensioning strategy, a good understanding of its fundamental properties is essential. We then discuss related work and present the processor sharing models that are employed for network dimensioning.

4.2 Transmission Control Protocol (TCP)

The *Transmission Control Protocol* (TCP) is the main protocol for data transmission in the Internet. It is a connection-oriented transport protocol, which assures reliable and ordered data delivery. Originally specified in [Pos81], many options and features have been proposed and added over the years in order to improve its performance and to adapt it to ever-changing environments. As a consequence, a variety of versions and implementations exist with their differences being mostly in the area of congestion control [Jac88, BOP94, FF96, APS99, FH99, Flo01]. The following description is for *NewReno*-type implementations, which are most common in today's IP networks [PF01]. However, as our dimensioning procedure assumes a macroscopic view of the TCP protocol, we are only interested in the fundamental behavior. Therefore, we do not go into more detail than necessary to understand the consequences for the dimensioning model. Comprehensive introductions of the TCP protocol can be found in [Ste94] or [Com00].

4.2.1 Maximum Segment Size and Maximum Transfer Unit

The *Maximum Segment Size* (MSS) is the largest amount of payload that a TCP instance packs into one IP packet [Pos83]. The MSS is negotiated at connection setup and should be chosen in a way that IP datagram fragmentation is avoided in the Internet. For this, the sum of MSS plus IP and TCP headers (together 40 bytes header) has to be smaller than the smallest *Maximum Transfer Unit* (MTU) of all physical interfaces on the path between the sender and the receiver. Ethernet for example has an MTU of 1500 bytes, which gives an MSS of 1460 bytes. To determine the minimum MTU on a path, a TCP sender can perform MTU discovery where intermediate networks are probed for their minimum MTU. Two MSS default values, which are often used in the Internet, are 512 and 536 bytes.

Consequences for Network Dimensioning

The values for MSS and MTU need to be considered during the network dimensioning process as they determine the amount of overhead, which TCP/IP introduces. As elastic traffic is often characterized on file level, every TCP flow correlates to the transfer of an individual file or web object. Therefore, the payload traffic volume can be derived from file size characteristics. However, the gross traffic, i.e., the amount of traffic, which is actually sent over the network, additionally includes TCP and IP headers. For a packet with payload length L_P the overhead factor is defined as $\frac{L_P+40\ bytes}{L_P}$. From [TMW97] it can be concluded that about 50% of all IP packets, which carry payload data, have a size of either 552, 576, or 1500 bytes (with roughly equal shares of packets with 552/576 bytes and 1500 bytes). Assuming that files are broken down into data chunks of MSS, we have overhead factors of 1.08, 1.07, and 1.03, respectively. Since smaller data objects or the last pieces of longer data transfers often form packets, which are shorter than MSS, the relative overhead is even a little higher.

4.2.2 Connection Establishment and Termination

TCP establishes new connections by carrying out a *three-way handshake*. The connection-requesting instance (usually a client) sends a SYN segment to the server. The server responds to the request by sending its own SYN segment and at the same time acknowledging the SYN of the client. To conclude connection setup, the client has to acknowledge the SYN of the server. Thus, the three-way handshake procedure involves three packets (SYN – SYN/AKC – ACK), each 40 bytes long.

After the data phase, the TCP connection has to be terminated again. To close a connection, an instance sends a FIN segment indicating that it does not have any more data to transmit. The other side acknowledges this FIN segment by sending an ACK. Since TCP connections are full duplex, one FIN – ACK sequence terminates the connection only in one direction (this is called *half-close*). To shut down the connection in both directions, the instance that has not yet sent a FIN has to do so (which, of course, has to be acknowledged by the other side). We see, for connection termination, a total of four segments have to be exchanged (FIN – ACK – FIN – ACK).

Consequences for Network Dimensioning

The connection establishment process defers the beginning of data transmission by about one round-trip time. For small file sizes, this extra time makes a difference in the achieved throughput. However, for longer data transfers, the duration of the connection setup procedure becomes negligible. In any case, the extra time can easily be taken into account during the network dimensioning process if an estimate for the round-trip time can be made. One has to dimension for a slightly higher throughput than actually demanded, in order to compensate for the lost time at the beginning of a transaction. However, with increasing file size, this additional bandwidth is negligibly small.

The connection termination procedure has no effect on the achieved throughput since it occurs after all relevant data have been sent. Therefore, it does not have to be considered any further.

4.2.3 Sliding Window Mechanism

TCP employs a *sliding window mechanism*. During connection establishment, a receiver might advertise a maximum amount of data, which it is able to hold in its receive buffer (this corresponds to the *maximum window size*, *wnd*). During transmission, the sender can only send this much data without having received an acknowledgement. This avoids buffer overflow on the receiver side, which could happen when a fast sender transmits data to a slow receiver. The window mechanism enables the receiver to actively control the amount of data that it receives by temporarily setting the window size to zero every time the buffer fills up.

To allow continuous sending without any unnecessary waiting times, the window has to be equal to or larger than the *bandwidth-delay product* of the path between the two TCP instances. This product describes the "capacity" of the network and is computed by $C \cdot RTT$. Parameter C is the available bandwidth in the network, measured in *bps*, and RTT is the round trip time between the sender and the receiver in seconds, i.e., the time period between sending a datagram and receiving the corresponding ACK.

Consequences for Network Dimensioning

The sliding window mechanism might have the effect that a sender cannot fully utilize the bandwidth that is assigned to it. Figure 4.1b illustrates what happens if the advertised window size is not chosen large enough. After having concluded the start-up phase (see next paragraph), the sender transmits with a window size, which does not allow continuous transmission. There are periods, when the window is exhausted and the sender has to wait. Therefore, the obtained throughput would be smaller than anticipated. However, as this service degradation would be due to a poorly configured end system, we do not consider it an issue, which a network service provider would have to take care of. Anyway, the processor sharing model, which will later be proposed for network dimensioning, would actually be able to capture the effect of a reduced maximum transmission rate due to the window mechanism.

4.2.4 Slow Start Mechanism

The *slow start mechanism* is a means to avoid network congestion after a new TCP connection is established and the sender starts transmitting data. Instead of utilizing the total advertised window size *wnd* right away and, thus, injecting a larger amount of data into the network, the

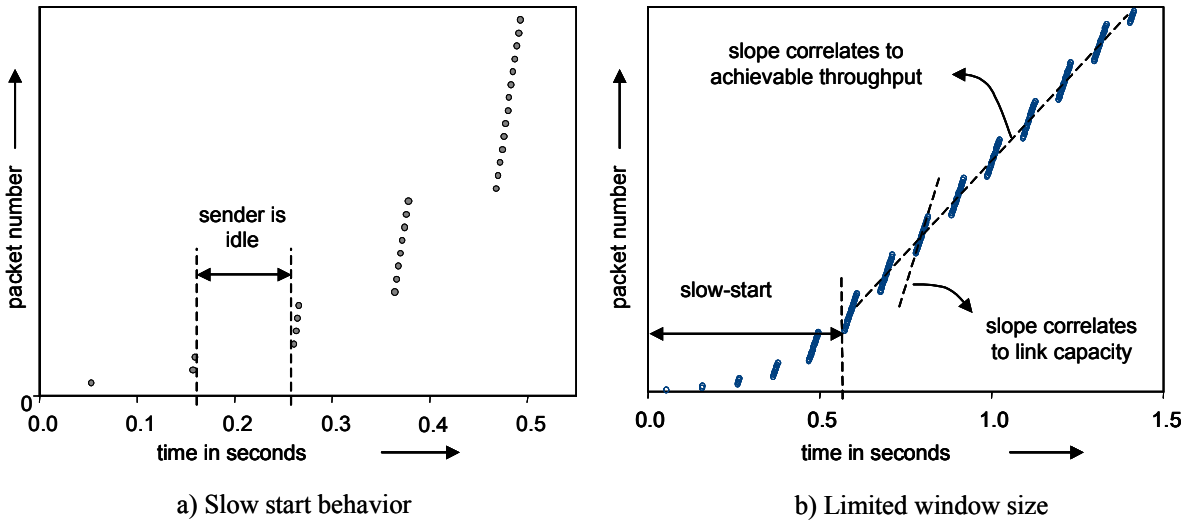


Figure 4.1: Slow start behavior and window mechanism of TCP

sender starts out slowly. To do so properly, it maintains another window parameter, called *congestion window cwnd*, which is initialized to one segment. At any time, the sender can only have up to the minimum of *wnd* and *cwnd* bytes unacknowledged. At first, the sender transmits only one segment. For each received acknowledgement it increments the congestion window by one segment. This leads to an exponential increase of *cwnd* until *wnd* is reached. Figure 4.1a illustrates the startup process by showing the sent packet number over time. The propagation delay on the link between the two TCP instances is 50 *ms* (round trip delay around 100 *ms*).

Consequences for Network Dimensioning

The slow-start mechanism of TCP slows down the transmission process and, thus, degrades the observed QoS. Especially, in networks with longer round-trip times this effect becomes noticeable. Therefore, if longer round-trip times are probable, it is advisable to take the slow-start mechanism into account and compensate the reduction of throughput through slight increase of bandwidth assignment. A possible solution is proposed in Section 4.4.5.

4.2.5 Congestion Avoidance Algorithm

While the slow start mechanism is used to avoid traffic overload at the beginning of a new connection, the *congestion avoidance algorithm* applies to congestion, which arises during the transmission phase. Although slow start and congestion avoidance are two independent mechanisms, they are usually implemented together and make up the congestion control mechanism of TCP.

The TCP protocol recognizes traffic congestion and packet losses either through a timeout or through reception of duplicate acknowledgements. In the first case, the retransmission timer, which is set for every sent packet, goes off. Initially the timer value is set to 1.5 seconds. During the transmission process, the round trip time is continuously estimated and the timeout value is adapted appropriately. In the case of duplicate acknowledgements, the

sender might conclude that packets have been lost. A duplicate acknowledgement is sent by the receiver when it receives an out-of-sequence packet. This might indicate either a loss of packets or just an overtaking of a packet by others that have been sent later. Therefore, the sending TCP instance does not react when only one duplicate acknowledgement is received. Only after several duplicate ACKs (usually three) the data is retransmitted.

The congestion avoidance algorithm and the slow start mechanism require two parameters, the congestion window *cwnd* (see slow start mechanism) and a *slow start threshold ssthresh*. Furthermore, the advertised window size (*wnd*) has to be considered. The algorithms together work as follows:

1. The window size at any time is the minimum of *wnd* and *cwnd*. The TCP output routine never sends more than that amount of data without receiving the appropriate ACKs.

2. Initially, *cwnd* is set to one segment and *ssthresh* is set to 65535 bytes (maximum possible window size).

3. With every received ACK *cwnd* is incremented by one segment size (slow start) until *cwnd* is larger than *wnd*.

4. When congestion occurs, *ssthresh* is set to one half of the current window size (either *cwnd* or *wnd*). If a packet loss was indicated by a timeout, *cwnd* is set to one segment size (slow start). If the packet loss was indicated by the reception of three duplicate ACKs, *cwnd* is set to *ssthresh* plus 3 times the segment size (fast retransmit).

5. When new data is acknowledged, *cwnd* is increased again. The increments, however, depend on whether $cwnd \leq ssthresh$ or $cwnd > ssthresh$. If $cwnd \leq ssthresh$, *cwnd* is increased by one segment size for each acknowledgement (slow start behavior with exponential increase). In case of $cwnd > ssthresh$, *cwnd* is only increased by $1/cwnd$ for every received ACK (congestion avoidance). This corresponds to an additive increase of about one segment size per round-trip time period.

The congestion avoidance mechanism leads to the characteristic behavior of *additive increase, multiplicative decrease* of a TCP source's transmission rate. Whenever TCP traffic traverses a bottleneck, where packets are lost, the sources start regulating their transmission rates by decreasing and increasing the window size. As a consequence, TCP transmitters do not send out data with a constant packet rate. It rather varies in order to gain a certain share of the bottleneck capacity.

Figure 4.2 illustrates the different congestion control mechanisms. The graph shows the evolution of the window size as well as the packet rate of a TCP sender for a scenario with one connection and one bottleneck link. The packet rate is normalized with regard to the bottleneck bandwidth.

At the beginning of the transmission process, the window is increased exponentially, leading to a send rate, which grows accordingly. As the packet rate is soon greater than the link's bandwidth, the buffer queue starts building up and overflow is inevitable. At time $t = 1.7$, a packet loss is detected through duplicate acknowledgements. The window size is halved and the respective segment is retransmitted. However, the packet is again lost leading to a timeout at $t = 2.5$. At this point, the window size is reduced to 1 and the slow start phase is entered. When the window size reaches *ssthresh* (which was set to 20 after the timeout),

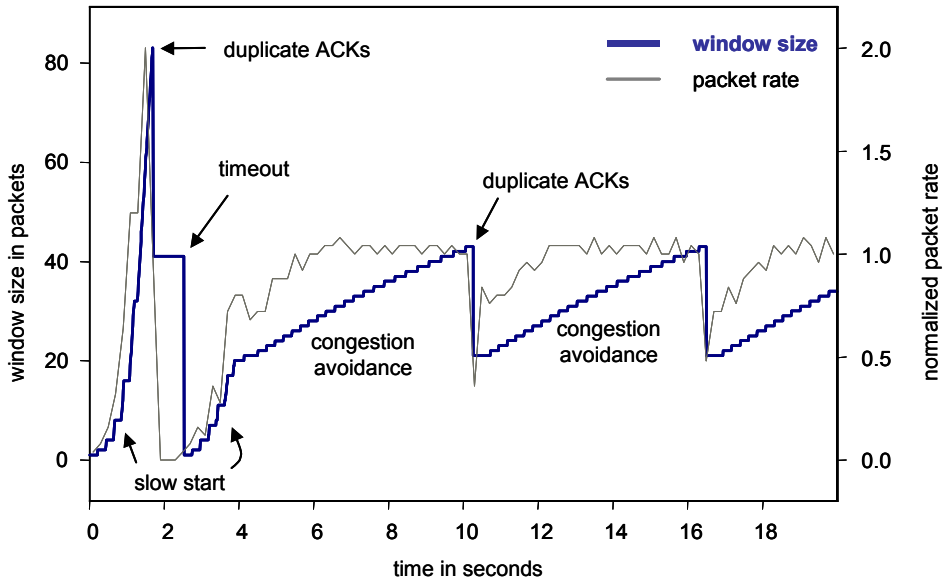


Figure 4.2: Congestion control mechanisms of TCP

it is increased only in an additive way. The packet rate at the sender roughly follows the window evolution process. However, during the congestion avoidance phase the effective send rate can only be slightly higher than the bandwidth of the bottleneck link. The sender is allowed to output one data packet for each returning ACK (which corresponds to the bottleneck bandwidth) plus one extra segment per RTT. Due to the slightly higher rate, the buffer again starts filling up, leading to another packet loss at $t = 10.2$. This time retransmission is successful and the TCP sender performs fast retransmit right away.

Consequences for Network Dimensioning

The congestion avoidance mechanism is one of the fundamental causes of traffic "elasticity" as it forces TCP to adapt its send rate to the available bandwidth. Therefore, it is not possible to dimension a network for elastic traffic without taking into account this specific feature of TCP. Every model has to consider it in some way. In our case, the congestion avoidance algorithm together with the slow-start mechanism provide the basis for the proposed processor-sharing dimensioning model.

4.2.6 Delayed Acknowledgments

The principle of delayed acknowledgements is used to increase the efficiency of the Internet by reducing the number of ACKs being sent [Cla82, Bra89]. Before sending an acknowledgement, the receiver waits for a short time (however, not more than 0.5 seconds) to see whether more data packets are arriving. If so, it can acknowledge more than one packet with only one ACK packet. Especially during slow start phase this mechanism is somewhat contra-productive since the transmit window is not increased with every correctly transferred data packet, slowing down the exponential increase of the window size.

Consequences for Network Dimensioning

Delayed acknowledgements lead to smaller average throughput than expected since they extend the slow-start phase. Therefore, if we have a feeling of how many TCP implementations (market share) use this feature, we could consider it. The approach would be very similar to the one that we show later on for slow-start.

4.2.7 Nagle Algorithm

The Nagle algorithm, originally presented in [Nag84], is used to reduce the sending of small packets (so called *tinygrams*). If the amount of data, which is waiting in a TCP send buffer is less than MSS the data is held back until either more bytes have to be transmitted or all outstanding data has been acknowledged. Only then does the sender transmit the small packet. For bulk data transfers, this means that the last packet of a transmission process might be delayed until the second last packet has been acknowledged. However, since the Nagle algorithm might negatively interact with the delayed acknowledgement feature, it is often turned off.

Consequences for Network Dimensioning

The Nagle Algorithm defers the sending of the last packet by at most one RTT. If this extra delay makes a difference in the transmission process (e.g., because of mainly small file sizes), we can consider it in our computations (see section 4.4.5.2).

4.3 Related Work

In this section we give an overview of dimensioning methods for elastic traffic. We first present important analytical models and methodologies, which establish a correlation between the throughput of TCP flows and the characteristics of the traversed links. Not all of the presented models are directly applicable to link or network dimensioning. They are often based on a very detailed view of TCP and, therefore, can only be applied to individual flows. However, we still consider them noteworthy at this point as they describe the behavior of TCP analytically and, thus, provide insight into performance aspects of TCP. In the second part we present models, which mostly omit the detailed investigation of TCP. They rather exploit its fundamental behavior of bandwidth sharing and utilize this property for network dimensioning.

4.3.1 Performance Models for Persistent TCP Flows

All approaches in this paragraph investigate stationary performance issues of individual TCP connections, which have an unlimited amount of data to send (*persistent sessions*).

Performance of Individual Flows

The following models infer transfer characteristics by looking at TCP's congestion control mechanisms. From the evolution of the congestion window size, analytical throughput formulae for individual flows are derived. The window determines how many packets a TCP

instance can send out without having received any corresponding acknowledgements. As described above, it is adapted by TCP during the transmission process, in order to avoid network congestion. Assuming that the round trip time RTT is long in comparison to serialization times, TCP approximately sends out rounds of packets in equi-distant time intervals of length RTT. As the number of packets per round is limited by the window size, the average send rate of a TCP flow can be derived from the distribution of the window size. The presented models differ in the exact TCP mechanisms, which are incorporated, and the network characteristics, which trigger the TCP mechanisms. All models include the packet loss rate p as one network characteristic, which greatly influences the window evolution process.

Ott, Kemperman, and Mathis [OKM96] consider the basic congestion avoidance algorithm with additive increase and multiplicative decrease behavior, which is triggered by random packet losses with probability p. Window reductions to 1, which are a reaction to timeouts, are neglected. Thus, the model best fits the TCP SACKS implementation [MMFR96]. Furthermore, they assume that the maximum window size is infinite. For small packet loss probabilities p, they derive an expression for the average throughput of a TCP session, which is a function of MSS, RTT, and p. In [MSM97] the applicability of this formula is demonstrated for various situations by simulation as well as by measurements. A generalization of the model for variable packet loss rates is presented in [MO99].

Padhye et al. [PFTK98, PFTK00] also investigate the performance of a single TCP flow, which is subject to random packet loss. However, they do not only model the window decrease due to duplicate acknowledgements. They also take into account the timeout mechanism and the effects of limited maximum window size. Furthermore, they are able to capture the delayed-acknowledgement feature where b packets are acknowledged at a time. This has an effect on the additive increase behavior in between two packet loss instances. They present several formulae for send rate and throughput of different TCP realizations. The calculations require knowledge of the parameters RTT, b, p, the timeout value T_0, and the maximum congestion window W_m. The formulae are validated by measurements.

Lakshman et al. compare the performance of Tahoe- and Reno-style TCP implementations when sending data over bottleneck links. In [LM97] the stationary window evolution process is investigated for single as well as multiple flows traversing networks with high bandwidth-delay products. An important difference to the preceding approaches is that the network is explicitly represented through a bottleneck with capacity C and buffer size B. Packet drops are not only introduced randomly, but are rather deterministic due to buffer overflow, which is a consequence of increasing window sizes. Thus, the model incorporates the feed-back loop between the bottleneck parameters and the window evolution process. However, the resulting throughput formulae are more complex. In [LMS97] and [LMS00] the models are extended taking into account asymmetric networks, i.e., networks where the bandwidth of the reverse (acknowledgement) path is considerably lower than the bandwidth of the forward (data) path.

Altman, Avrachenkov, and Barakat [AAB00b, AAB00a, AABD00] present transmission rate formulae for TCP, taking into account the congestion avoidance mechanism, limited window size, and coarse timers. The reduction of the congestion window is assumed to be triggered by random packet losses. However, they argue that it is not sufficient to only consider deterministic or independent random packet losses. As it is quite common in the

Internet that losses occur in a bursty way, the characteristics of the loss process have to be taken into account in order to evaluate the performance of TCP correctly. Therefore, they explicitly model the distribution and the correlation structure of inter-loss times, allowing them to investigate several types of loss processes.

Network Scope

So far, the presented work was only concerned with individual flows. The following publications propose methods to evaluate the performance of a network for a given number of persistent TCP flows. These methods could be used within a network dimensioning algorithm (e.g., genetic algorithm) to evaluate a potential solution.

Gibbens et al. [GSE+00] define a fixed-point model, where the transmission rates of the individual TCP sources are dependent on the packet loss probabilities at the links along the respective paths. On the other side, the loss probabilities are influenced by the transmission rates of the individual sources. Thus, the individual flows are interrelated due to the packet loss rates, which they experience at shared links. In the paper, Padhye's formula is taken to establish the dependency of the rates on the loss probabilities. In order to derive the packet loss probabilities, a router is modeled as an M/M/1/K queue.

Roughan, Erramilli, and Veitch [REV01] present a similar fixed-point methodology to evaluate the performance of TCP flows in a network scenario. They provide a general framework where they do not specify the exact form of the dependencies between packet loss and throughput. However, in their examples they use a short form of Padhye's result for the dependency on TCP level and a simple bufferless excess flow scheme for the computation of packet losses on links.

4.3.2 Dimensioning Models for Non-Persistent Flows

The models in the preceding section capture the stationary behavior of TCP sources, which have a large amount of data to transmit. However, in practice this is usually not the case as many TCP connections transfer only single files or web documents and are closed after the transaction. Therefore, if one is interested in dimensioning aspects, it is necessary to also account for the arrival and termination of TCP flows.

Heyman, Lakshman, and Neidhardt [HLN97] first presented a bandwidth-sharing model taking into account non-persistent elastic flows. Their approach assumes a fixed number of users who alternate between two states: downloading a document (ON-time) and thinking (OFF-time). During the download phase, a single document is transmitted from the web server to the client. As all users share a common bottleneck link, the rate of service during ON-time depends on the number of simultaneous downloads. The overall system is described by a modified version of the Engset queueing model where rate reduction occurs instead of call blocking. The achievable throughput follows from the steady-state probabilities and the corresponding, state-dependent download rates.

Berger and Kogan [BK00, BK99] assume also idealized bandwidth sharing of TCP flows and investigate a single bottleneck by means of a closed-queuing network with two nodes. The first node, an infinite-server node, holds all inactive jobs. When a job becomes active, it

is served at the second node in processor sharing fashion. After receiving the total required service time, it returns again to the infinite-server node where it will remain for a random period of time. Thus, the model is very similar to the approach of Heyman, Lakshman, and Neidhardt. However, Berger and Kogan analyze the closed-queuing network for heavy traffic and derive simple dimensioning rules for different QoS measures.

Kherani and Kumar [KK00] explicitly consider TCP's congestion avoidance algorithm with additive increase and multiplicative decrease. They investigate the window evolution process and directly translate it into a fluid-flow rate process, taking into account non-persistent TCP flows. Flows represent file transfers, which arrive according to a Poisson process and which have an exponentially distributed size. It turns out that the specified rate process is Markov regenerative and that the steady-state probabilities and consequently the average throughput measure can be computed. However, since the model is very complex to analyze, the authors compare the results with simple processor sharing and find them to be in fair agreement if the processor sharing model assumes a link capacity of about 75% of the actual capacity.

Approaches based on Processor Sharing

Although the detailed coverage of processor sharing in the context of network dimensioning is deferred to the next section, we still present here a short overview of work done in this area. Several groups of researchers promote processor sharing as a tool for link dimensioning, which is applicable in different situations and scenarios.

Roberts suggested the use of processor sharing for dimensioning in [Rob97]. Massoulié and Roberts propose to carry out call admission control for elastic traffic in order to guarantee a minimum achievable throughput for all accepted flows [RM98, MR99]. The link capacity, which is required to accommodate the elastic flows, can again be calculated with the processor sharing model. In [BR00] and [OR00] dimensioning issues for Differentiated Services are discussed. Van den Berg et al. propose formulae for a similar context in [vdBvdMG+01].

Lindberger [Lin99] also presents the processor sharing model as an appropriate means of dimensioning. He derives the M/G/R PS model for cases without flow blocking and introduces the "delay factor" as a QoS measure for elastic traffic (for a more detailed discussion see section 4.4). Furthermore, he proposes approximations for the combined dimensioning of stream and elastic traffic.

Núñez Queija, van den Berg, and Mandjes investigate the problem of service integration in more detail [QvdBM99a, QvdBM99b]. They derive blocking probabilities and mean transfer times for different link sharing strategies between stream and elastic traffic flows. The computations for elastic traffic are based on processor sharing, while for stream traffic a blocking model is used. However, the integrated models are too complex to be applied to dimensioning with a network-wide scope.

Beckers et al. [BHKvdM01] validate the accuracy of the processor sharing model through packet-level measurements in an ADSL access network. They find that the measured transmission times are mostly in accordance with the theory. However, they also point out shortcomings of the model, which are mainly attributed to round trip times. In order to reduce the error, they suggest to explicitly consider the round trip times in the formula. This approach is

very similar to our solution, which will be presented in section 4.4.5 and which was published in [RPBP00b].

4.3.3 Conclusion

The detailed models for persistent flows provide good insight and understanding of TCP's behavior and characteristics. They allow thorough performance investigations in view of different scenarios and environments. However, due to their complexity they are not very practical for the purpose of network dimensioning. For network dimensioning a more macroscopic approach is necessary. This is why the processor sharing model has become very popular. It hides the details of the TCP protocol, requires only a few input parameters, and can be solved quickly. Overall, it provides a simple, yet powerful theoretical framework.

However, while numerous variations of the basic formula have been proposed, not much work was published concerning model validation. Therefore, we dedicate a considerable portion of our work on simulating links and networks in order to verify the applicability of the model and to identify possible implications. As we apply processor sharing not only for link dimensioning, but extend it to the scope of network structures, validating the model becomes even more important. First simulation results were published in [RPBP00a]. As mentioned above, for longer round trip times the basic model becomes inaccurate. Therefore, we propose an extended model, which can be used if necessary [RPBP00b].

4.4 Processor Sharing Models for Elastic Traffic

Processor sharing (PS) models were originally developed to evaluate multi-tasking schemes on computers with a single processor [Kle76]. The computation resource is shared among several jobs by specifying time slots, during which the processing power is exclusively utilized by one of the customers. Typically, the processor time is assigned in a round robin manner. If the time shares are small enough, the computation of the individual jobs appears to be quasi-simultaneous. The limit of the time-sharing process with the duration of the individual slots approaching zero is denoted as processor sharing. Then, all customers are theoretically served simultaneously by granting them a certain rate of computation instead of individual time slots.

Much work has been published, investigating the characteristics and the performance of such systems (see for example [Coh79, Ott84, Yas87, Yas92, ZB98, BBJ99] and references therein). One performance measure, which is often considered in the context of processor sharing and which is specifically interesting for link dimensioning, is the sojourn time. It denotes the expected time, which a job spends in the system until it is fully serviced . It is clear that this time depends greatly on the number of other jobs, which are served during the same time period. Each new customer that enters the system reduces the available rate. A finished job, on the other side, increases it again for the remaining ones.

4.4.1 Rationale behind the Use of Processor Sharing Models

For the purpose of link dimensioning, the processor sharing model has to be interpreted differently. Instead of dividing a computational resource between several jobs, the bandwidth of a link is now shared by a number of elastic flows. What was referred to as a job or customer

in the original sense, denotes now a single transaction process, i.e., the transmission of an individual file by one TCP connection.

As we have explained above, TCP controls the transmission rate of a sender by adapting the congestion window size. Assuming for the moment that TCP's feedback and control mechanism is perfect and absolutely fair, the bandwidth of a common bottleneck link would be shared equally among active connections. Whenever a new TCP instance starts sending data, all TCP connections, which are already in progress, instantaneously reduce their rate in order to provide a fair share for the new one. If a TCP sender is finished, the remaining connections increase their rates so to fully utilize the available bandwidth again. Thus, a common bottleneck link, which is traversed by several elastic flows, can be considered a processor sharing system. However, it is important to understand that processor sharing in this context refers to the connection or file transfer level. There is no processor sharing at packet level. A router still forwards the packets belonging to the individual flows in a first-come first-serve fashion. Only on the coarser-granular connection level it appears as if the flows are passing through the link (and the router) simultaneously with the available portion of bandwidth. As the time scales of the individual file transfers are (at least for longer files) several orders larger than the individual packet delays, it is not relevant how fast every packet is transmitted from the source to the destination node. Ultimately, it only matters, how fast the whole file is moved from the sender to the receiver.

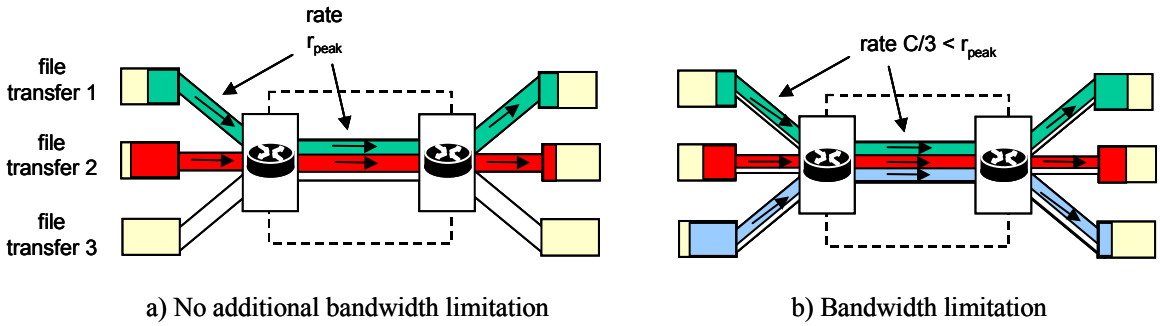


Figure 4.3: Illustration of rate sharing

Since congestion control is performed by TCP on an end-to-end basis, we can further assume that the packet rate of a TCP connection is the same along the whole path between the sender and the receiver. Thus, any rate limit along the path of a TCP connection effectively limits the bandwidth, which this connection can utilize on any other link. This is illustrated in Figure 4.3, which shows two snap-shots of a network scenario with a single shared link. The maximum achievable rate of a single TCP source is limited to the capacities of the access links (r_{peak}). For the capacity C of the shared link we assume here that $2 \cdot r_{peak} < C < 3 \cdot r_{peak}$. If only two TCP connections are active (Figure 4.3a), each TCP sender can transmit data with the maximum possible rate of r_{peak}. Since the two flows are not able to fully utilize C, the link does not affect the transmission processes. However, if a third flow becomes active, the sum of all peak rates exceeds C, and the shared link becomes a bottleneck. Instead of sending packets with rate r_{peak}, the sources lower their transmission rates to $C/3$ (Figure 4.3b). In general, if there are n elastic flows sharing a common link with capacity C, the send rate r of each flow is

$$r = \min\left(C/n, r_{peak}\right)$$

If the rates of the flows are not limited or if $r_{peak} \geq C$, then only C/n is relevant. This is the case, which corresponds to simple processor sharing, where the total rate is always divided up equally among all active flows. As long as there is at least one flow in the system, the shared link is always fully utilized. For $r_{peak} < C$ this is not the case, since it takes a certain number of active flows to fully load the link.

By applying the theory of processor sharing, we can investigate dynamic scenarios where new flows arrive at the system and finished flows leave it. The theory allows us to derive the dependency between the expected transmission time, i.e., the sojourn time, and the amount of data, if the peak rate of the individual connections and the capacity of the shared link are given. The underlying assumptions are described in the next paragraph.

4.4.2 Assumptions

The PS models are based on IP traffic characterization at flow or connection level. We assume that every elastic traffic flow is principally related to a single file transfer. However, considering the World Wide Web as the main application, one traffic flow might also represent (at least approximately) a number of different web objects downloaded from the same server. Since HTTP/1.1 [FGM+99] allows the download of several web objects within one TCP connection, individual transmission processes look like one large process. The size of the virtually large document is equal to the sum of the individual object lengths. Measurements of web traffic have shown that the amount of transmitted data shows heavy-tail characteristics [Cha00]. This means that while most of the transfer processes are short, there exist some extremely long ones. Most likely, this effect is intensified by popular peer-to-peer file sharing applications where large audio and video files are exchanged. Thus, it can be expected that in the future the characteristics of heavy-tailed file length distributions will prevail.

It has been observed that interarrival times are also sub-exponentially distributed (e.g., Weibull) [Fel00]. However, as these characteristics are quite difficult to deal with analytically, the flow arrival process is typically modeled as a Poisson process.

In summary, the theoretical models are built on the assumptions that new file transfers start according to a Poisson process and file sizes are generally distributed, e.g. modeled by a hyper-exponential or Pareto distribution. The presented formulae are insensitive to the service time distribution [Lin99, BPRR01]. Thus, elastic traffic is characterized by the parameters *file arrival rate* (or *flow arrival rate*) λ_e and *mean file length* l_e, or by the product of these two values, the average traffic volume a_e. The service quality of elastic traffic transmission is specified through one of the two notions of throughput presented earlier. Furthermore, a generic measure, the *delay factor*, will be introduced, which can be used to quantify the QoS instead.

4.4.3 M/G/1 PS Model

The M/G/1 Processor Sharing model describes the situation where every source has the ability to fully utilize the link rate C in times when no other flow is present in the system (i.e., $r_{peak} \geq C$). For our TCP scenario, this means that there do not exist any bottlenecks outside the system, which would limit the potential transmission rate of individual flows.

When n flows are active at a certain time, they are simultaneously served, each receiving a rate of C/n.

It can be shown [Kle76] that the sojourn time $T(x)$, i.e., the expected time until a file of length x (including overhead) is fully transferred, is given by

$$T_{M/G/1}(x) = \frac{x}{C \cdot (1-\rho)} = \frac{x}{C - a_e} \tag{4.1}$$

Parameter ρ denotes the utilization of the link, which follows from the mean service arrival rate λ_e and mean file size l_e or from the average traffic volume a_e by $\rho = \lambda_e \cdot l_e/C$ and $\rho = a_e/C$, respectively. In practice it is often easier to predict the average traffic volume instead of arrival rates and file lengths.

Note that x/C represents the time of transferring a file of length x if no other flows are active. The existence of other flows throughout the course of the transmission process increases the average transfer time by a factor of $1/(1-\rho)$, which is often referred to as *delay factor* f. Note that the delay factor is always larger than 1 for $\rho > 0$.

4.4.4 M/G/R PS Model

In cases where the transmission rates of TCP connections are limited to a certain peak rate r_{peak} with $r_{peak} < C$, the M/G/1 PS model can be interpreted as an M/G/R PS model. A shared link appears like a system with $R = C/r_{peak}$ servers. Processor sharing comes only into play when more than R flows are active. For less than R active flows, each one can be served without rate reduction imposed by the shared link.

For an M/G/R-PS model where $R = C/r_{peak}$, the expected sojourn time (or transfer time) $T(x)$ for an amount of data of size x is given by [Lin99]:

$$T_{M/G/R}(x) = \frac{x}{r_{peak}} \left(1 + \frac{E_2\left(R, R\rho\right)}{R(1-\rho)}\right) = \frac{x}{r_{peak}} \cdot f_R \tag{4.2}$$

Consequently, the throughput D during the file transfer phase is:

$$D = \frac{x}{T(x)} = \frac{r_{peak}}{\left(1 + \frac{E_2(R,R\rho)}{R(1-\rho)}\right)} = \frac{r_{peak}}{f_R} \tag{4.3}$$

Here, ρ denotes again the utilization on the link ($\rho = \lambda_e \cdot l_e/C$ with flow arrival rate λ_e and average file size l_e). E_2 represents Erlang's second formula (Erlang C formula) with $A = R\rho$ (equation 4.4). The given equations are only exact, if R is an integer value. However, for fractional R we can use a continuous approximation for the Erlang function [FK78].

$$E_2(R, A) = \frac{\frac{A^R}{R!} \cdot \frac{R}{R-A}}{\sum_{i=0}^{R-1} \frac{A^i}{i!} + \frac{A^R}{R!} \cdot \frac{R}{R-A}} \tag{4.4}$$

Since the transmission rate is limited to r_{peak}, the minimum time for transferring a file of size x is x/r_{peak}. This duration is achieved, if at most R sources are sending data. Every additional flow, which becomes active during the transfer process, takes away bandwidth and, thus, delays the transfer. Therefore, the expression in the parentheses of equations 4.2 and 4.3 is again called *delay factor* f_R:

$$f_R = 1 + \frac{E_2\left(R, R\rho\right)}{R(1-\rho)} \tag{4.5}$$

The delay factor quantifies the expected prolongation of the transfer or the reduction of the effective throughput. As a flow can never achieve a throughput, which is higher than its peak rate, f_R is always larger than 1.

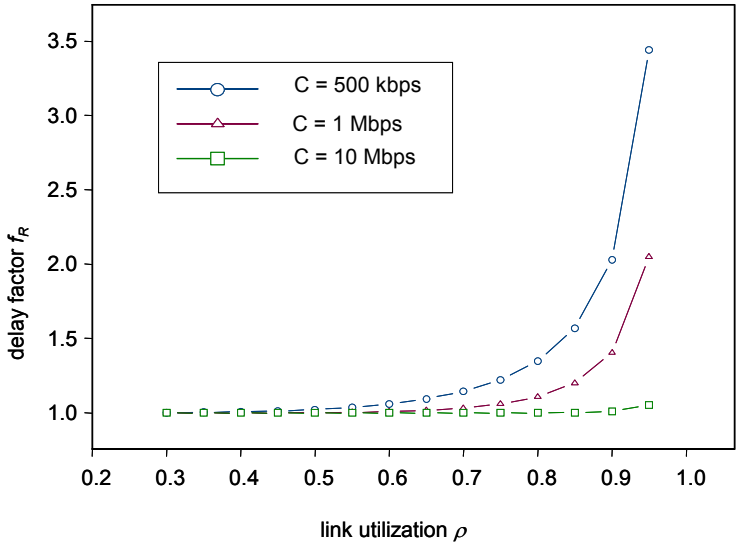


Figure 4.4: Delay factor over link utilization (r_{peak} = 64 kbps)

Figure 4.4 shows the delay factor as a function over link utilization for different values of the shared link capacity. The peak rate r_{peak} is 64 kbps. For small ρ the delay factor is very close to 1. This means that all connections receive an average rate, which is almost equal to their peak rate. For higher utilization values the delay factor increases, going to infinity for $\rho \to 1$. From the graph we can see that f_R reflects the economy of scale characteristic. Links with larger capacities (i.e., higher R) can be higher utilized than smaller ones while still achieving the same delay factor. Furthermore, the sensitivity of the delay factor regarding traffic variation increases drastically for smaller link capacities as the utilization approaches 1. Therefore, one has to be very careful when dimensioning links with lower capacities (e.g., access networks).

4.4.5 Extended PS Models

The preceding processor sharing models only consider the bandwidth sharing property of TCP, which they assume to be ideal, and ignore any network characteristics. However, in practice TCP flows are not necessarily able to utilize their fair share of available bandwidth. TCP's effectiveness is determined by the algorithms introduced earlier, which are affected by network conditions like round trip times and packet losses. Especially when round trip times are longer, TCP connections spend a considerable amount of their overall transmission time in the first slow start phase. Short transactions might not even get past the slow start phase at all. As a consequence, the achieved throughput will be lower than the theoretically computed one. For network dimensioning this means that the calculated capacities would not be sufficient to deliver the desired QoS. Therefore, we suggest extending the pure M/G/1 PS or M/G/R PS models by some extra terms in order to improve the accuracy of the models.

Then, we are able to compute appropriate capacities in case that long round trip times lead to throughput degradation. Our extended model considers the slow start algorithm, the connection setup times, and the Nagle mechanism, which all are affected by long round trip times and which do not really comply with the idea of pure processor sharing.

4.4.5.1 Slow Start Extension

The slow start mechanism increases the sojourn time by not utilizing the available bandwidth at the beginning of each transmission process. To investigate the impact, we differentiate between two states: a start-up state and a fair-sharing state. The first state indicates that the send window does not yet allow to achieve the expected bandwidth share since it is still too small. In the fair-sharing state, the source can utilize the designated bandwidth.

In order to derive an approximation for the expected transaction time, which includes the extra delay due to slow start, we make following assumptions:

- The total amount of data x is transferred in packets of size MTU.
- No packets are lost before the connection has reached its fair-sharing state.
- The maximum window size of a TCP connection is large enough to cover the respective bandwidth-delay product of the network.
- At any time, most of the active connections have already assumed their fair-sharing state.

Let $C_{available}$ denote the designated bandwidth share, which a sender is supposed to utilize in average once its window is large enough. For a given link capacity C, peak rate r_{peak}, and traffic intensity a_e, this bandwidth is determined by $C_{available} = r_{peak}/f_R$. Parameter f_R is the delay factor of the M/G/R PS system, i.e., $f_R = 1 + E_2\left(R, R\rho\right)/R(1-\rho)$. In case of unlimited peak rates, $C_{available} = C(1-\rho)$. This approximation is justified if fair-sharing is the dominant state (fourth assumption). Then, the overall system still behaves like a processor sharing system.

We can now calculate the time, after which a sender is able to utilize its fair share. This is the case when the congestion window is large enough to cover the bandwidth-delay product, i.e., when it reaches w^* with

$$w^* = \left\lceil \frac{C_{available} \cdot RTT}{MTU} \right\rceil = \left\lceil \frac{r_{peak} \cdot RTT}{f_R \cdot MTU} \right\rceil \tag{4.6}$$

Assuming that a TCP sender starts transmitting with an initial window of 1, the size of the window increases exponentially with approximately 2^i where $i = 0, 1, 2, ...$ is the number of completed round trip time intervals. It takes n^* round trip times until the window is for the first time equal to or larger than w^*:

$$n^* = \lceil \log_2 w^* \rceil \tag{4.7}$$

The amount of data, which has been sent until the window reaches or exceeds w^*, i.e., when the sender can start utilizing its bandwidth share, is

$$x_{ss} = \left(2^{n^*} - 1\right) \cdot MTU \tag{4.8}$$

If the total amount of data x is smaller than x_{ss}, the sender never reaches the state of fair-sharing. Instead, all packets are sent out in the start-up state. The computation of the expected time to send all packets in this case is illustrated in Figure 4.5. The first term is approximated by the number of round trip times, which the sender spends mostly waiting for acknowledgements. Here we assume that the round trip time is the dominating term and not the time of sending out the rounds of packets. During this first period of time, x_{start} bytes are transmitted with

$$x_{start} = \left(2^{\lfloor \log_2(\lceil \frac{x}{MTU} \rceil) \rfloor} - 1\right) \cdot MTU \tag{4.9}$$

The second term considers the time of sending the remaining data with the available capacity. The available capacity is assumed to be the bandwidth, which the source obtains as a share of the overall capacity. Thus, the final time period is computed using the pure M/G/R PS formula for the remaining amount of data.

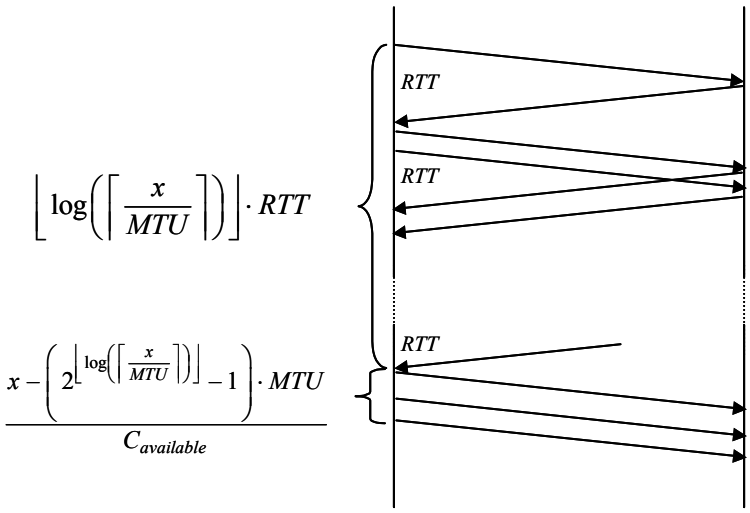


Figure 4.5: Start-up phase of TCP connection

In cases with more than x_{ss} bytes to send, we can also divide the overall transaction into two phases. The first phase lasts $n^* \cdot RTT$, and a total of x_{ss} bytes are sent. The duration of the second phase is again computed by the M/G/R PS formula taking into account the rest of the data.

In summary, we have following expected transmission times $T_{ss}(x)$ for files of size x (including overhead):

$$T_{ss}(x) = \begin{cases} \lfloor \log_2 \left(\lceil \frac{x}{MTU} \rceil \right) \rfloor \cdot RTT + T_{M/G/R}(x - x_{start}) & if \quad x < x_{ss} \\ n^* \cdot RTT + T_{M/G/R}(x - x_{ss}) & if \quad x \geq x_{ss} \end{cases} \tag{4.10}$$

For network scenarios with unlimited peak rates, $T_{M/G/R}$ is replaced by $T_{M/G/1}$. Figure 4.6 illustrates the effecs of long round trip times for a scenario with $r_{peak} = 64$ kbps, $C = 1048$ kbps, and $\rho = 0.86$. The graph shows the expected transfer time function over the amount of transferred data (file length). For $RTT = 0$ ms the regular M/G/R PS formula applies, which

produces a straight line (linear relationship between $T(x)$ and x, i.e., the same throughput for all file sizes x). For $RTT > 0$ ms, the curves deviate from this line due to the idle periods during the slow-start phase of each flow. These extra delays add up to a constant offset $(n^* \cdot RTT)$ as long as files are larger than x_{ss}. The offset increases for larger RTT. However, the final slope of the transfer time function is the same for all curves. It corresponds to the available bandwidth, which is proportional to the delay factor of the original M/G/R PS system and is computed as f_R/r_{peak}. Thus, for increasing f_R the functions become steeper as transfers take longer. Another important fact, which should be pointed out here, is that due to the constant offset the throughput is now dependent on the file size. For small files, the additional delay leads to great throughput degradations, while for large ones the offset is small in relative terms. Taking the example of $RTT = 1000$ ms, the effective throughput for file sizes of 20 kbytes reduces from about 53 kbps to 23 kbps. For files of size 150 kbytes the effective throughput is still roughly 45 kbps.

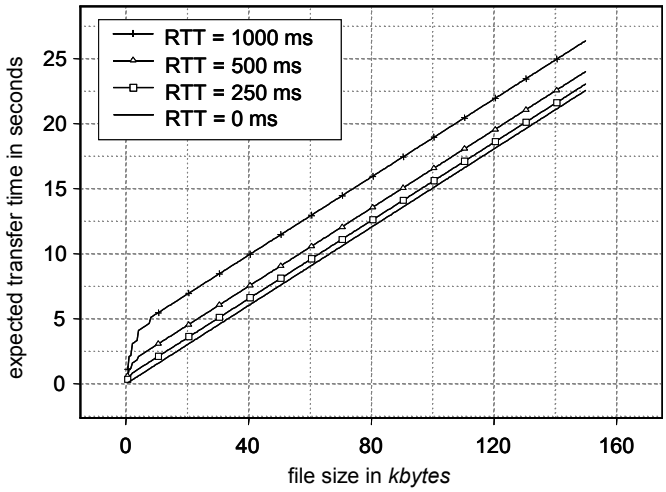


Figure 4.6: Expected transfer time function of Extended M/G/R PS

4.4.5.2 Nagle Extension

The processor sharing model can be extended to consider the Nagle algorithm. Since this algorithm delays the sending of the last packet, one round trip time RTT has to be added to the sojourn time:

$$T_{Nagle}(x) = T_{ss}(x) + RTT \tag{4.11}$$

4.4.5.3 Connection Setup Extension

The extra delay caused by connection setup and termination can be approximated by one time the expected round trip delay. This is about the time until the requesting instance can send data for the first time (right after sending the ACK segment, which concludes the three-way handshake). Since connection termination does not degrade the achievable throughput, we do not need to take it into account.

$$T_{cs}(x) = T_{ss}(x) + RTT \tag{4.12}$$

4.5 Dimensioning Procedures

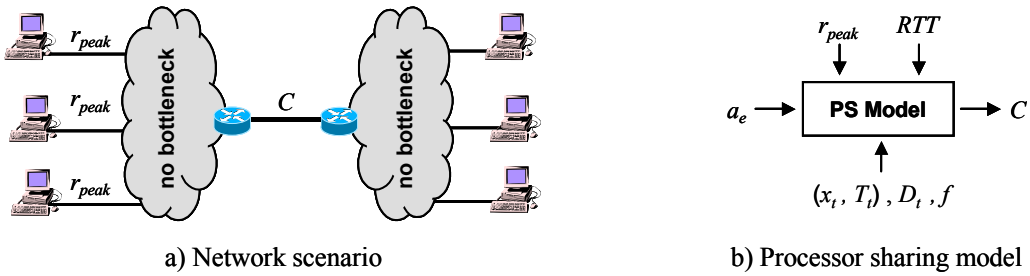
We now show how to utilize the presented PS models for dimensioning purposes. We assume that a traffic matrix is given, which specifies the amount of offered elastic traffic exchanged between all network nodes. The traffic includes the overhead, which can be approximated from the MSS distribution as discussed in section 4.2.1. Furthermore, due to fixed routing, we can determine the amount of elastic traffic on each link in the network.

In the following, we distinguish two dimensioning scenarios: dimensioning of individual links and of networks. In the first case, processor sharing applies directly, while in the second case it provides a good approximation. For both scenarios, the procedures are described.

4.5.1 Dimensioning of Individual Links

Let us first focus on the dimensioning of a single link as illustrated in Figure 4.7a. We assume that there are no other links in our network, which are shared by the traffic flows or which would influence the rate sharing behavior. Possible scenarios, to which this situation applies, are:

- Local area networks with directly attached computers: only one link exists in the network.
- Access networks with single concentration links: hosts are connected to an access router either directly or through dedicated lines. The router acts as concentrator towards a high-speed backbone network (see the discussion in the following paragraph).
- Networks with identifiable bottlenecks such as enterprise networks with medium-bandwidth WAN connections between remote sites: other links might exist in the network but their influence on the rate sharing behavior is considerably less than the bottleneck. Therefore, they can be neglected.



a) Network scenario b) Processor sharing model

Figure 4.7: Link dimensioning with processor sharing models

Figure 4.7b shows the input and output parameters of the processor sharing models. At first, we have to determine the amount of elastic traffic a_e, which traverses the inspected link. Furthermore, possible rate limitations confining individual flows have to be identified

and, optionally, round trip times can be estimated. For now, it is assumed that all flows experience the same peak rates and the same round trip times. At last, an appropriate QoS measure must be specified. As discussed earlier, this could either be the target throughput D_t, a desired transaction time T_t for a file size of x_t (including overhead), or simply a value for the desired delay factor f. In the first two cases, we define a desired QoS throughput of D_{QoS}, which is equal to D_t or to x_t/T_t, respectively. However, one should be aware that if the round trip times are taken into account, the sojourn time is not a linear function of the file size. The throughput rather depends on the file size. As a consequence, it is not possible to assume one general throughput value D_t. Instead, one has to specify the file size, for which D_t should be valid. This is again the same as giving the target values for T_t and x_t. The further procedure depends on whether or not there exist individual rate limitations.

Unlimited Peak Rate

For negligible round trip times, equation (4.1) applies. It can explicitly be solved for C, which gives

$$C = D_{QoS} + a_e \tag{4.13}$$

or in case the QoS requirement is expressed by means of the delay factor f

$$C = \frac{f}{f-1} \cdot a_e$$

If the effect of long round trip times should be taken into account, equation (4.2) with $T_{M/G/1}$ instead of $T_{M/G/R}$ has to be used. The procedure for finding the solution is analog to the one described in the following paragraph.

Limited Peak Rate

In cases with limited peak rates, the M/G/1 PS equations are solved first. If the resulting capacity C is smaller than or equal to the peak rate r_{peak}, the optimum solution is found.

If C is larger than r_{peak}, the optimum capacity can be computed by applying the M/G/R PS model. However, it is not possible to solve equations (4.2) or (4.10) explicitely for C. Therefore, we suggest to iteratively calculate the sojourn times for increasing capacities C. Starting from a value of a_e, C can be incremented in intervals of r_{peak} until the desired QoS is achieved, i.e., until $D \geq D_t$ or $T(x_t) \leq T_t$.

Different Peak Rates

If there exist different classes of customers with different peak rates, Lindberger [Lin99] suggests to use the weighted average of the various peak rates, i.e.,

$$r_{peak} = \frac{\sum_i a_{e,i} \cdot r_{peak,i}}{\sum_i a_{e,i}} \tag{4.14}$$

where $a_{e,i}$ denotes the traffic load of customer group i and $r_{peak,i}$ their respective peak rate.

The graphs in Figure 4.8 illustrate different characteristics of the dimensioning formulae based on M/G/1 PS and M/G/R PS. In Figure 4.8a the overdimensioning factor, i.e., the

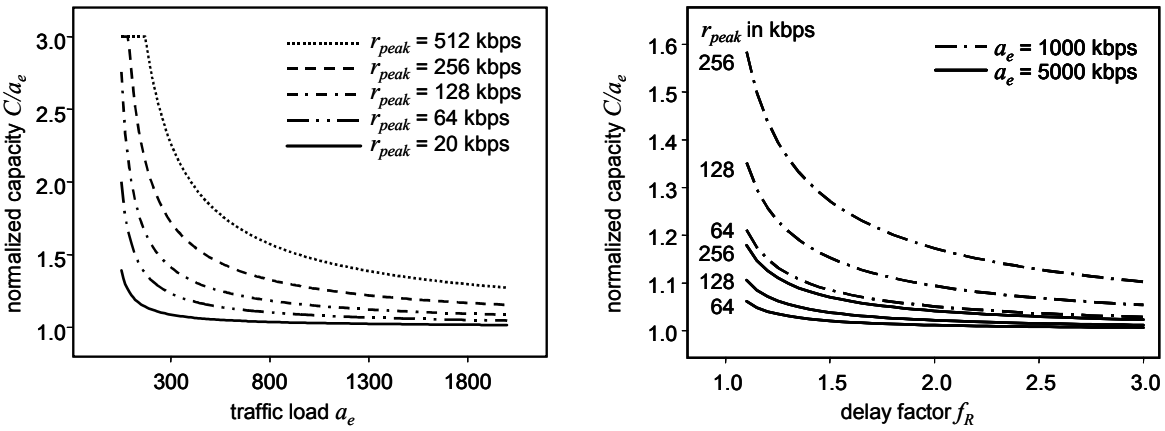


Figure 4.8: Characteristics of dimensioning formulae

required normalized bandwidth, is depicted as a function of the offered load. The desired delay factor was set to 1.5. Curves for various peak rates are shown. Consistent with the delay factor graph in Figure 4.4, we again observe a bundling gain for larger traffic aggregates as the overdimensioning factor approaches one for increasing traffic load. The discontinuity of the curve for $r_{peak} = 512$ kbps marks the transition point from the M/G/1 to the M/G/R PS model. For traffic loads smaller than 512/3 kbps the resulting link capacities do not exceed the peak rate and, thus, the M/G/1 PS model applies. The graph in 4.8b depicts the dependence of the required link capacity on the delay factor for two different aggregation levels and various peak rates. The influence of the peak rate on the link capacity is demonstrated in Figure 4.9 for different delay factors and an offered load of $a_e = 1000$ kbps. Larger peak rates require larger bandwidth values.

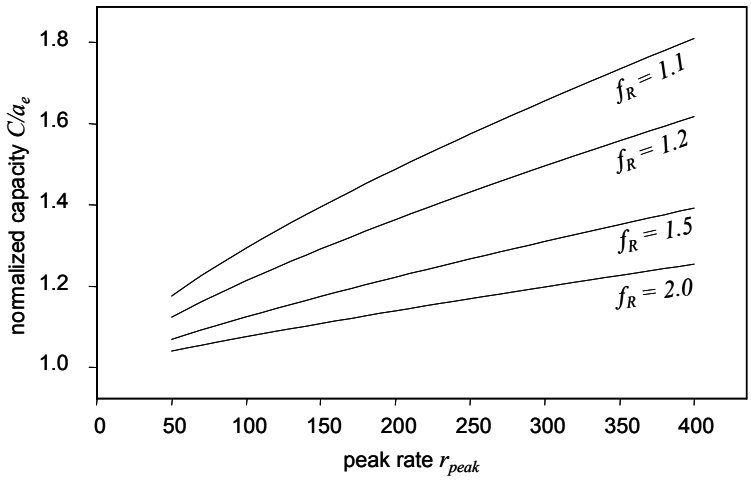
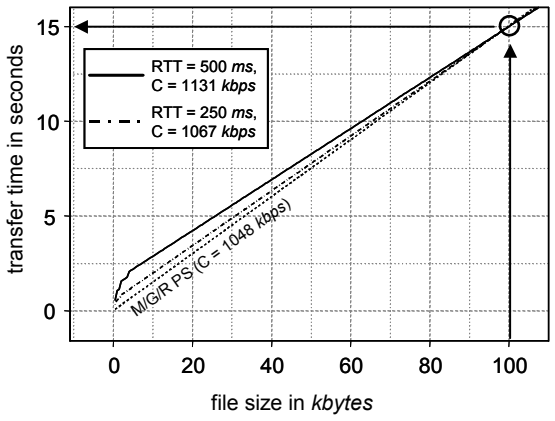


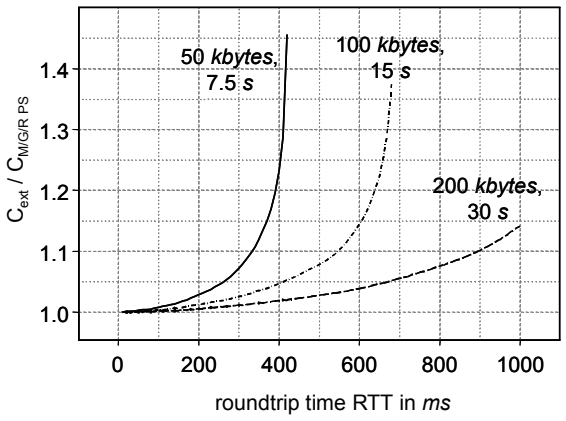
Figure 4.9: Capacity over peak rate ($a_e = 1000$ kbps)

Finally, we would like to outline the dimensioning process for scenarios with non-negligible

round trip times. At first, the desired QoS has to be specified in form of (x_t, T_t). Note that it is not possible now to specify just a value for the effective throughput as this will depend on the file length. In the example of Figure 4.10a an average transfer time of 15 seconds is requested for files of size 100 kbytes (this is marked by arrows in the graph). Applying the regular M/G/R PS model would result in a capacity of $C = 1048$ kbps. However, for $RTT > 0$, the initial slow-start delay has to be compensated by a slightly higher effective throughput (leading to a smaller slope of the transfer time function) in order to have the curve go through the point (100 kbytes, 15 s). For $RTT = 250$ ms, the required capacity is $C = 1067$ kbps, which needs to be increased to 1131 kbps if $RTT = 500$ ms. The dependency between required link capacities and round trip times is illustrated in Figure 4.10b. The capacity values are normalized with respect to the capacity of the corresponding M/G/R PS formula (which in the given scenario is 1048 kbps). Furthermore, the influence of the choice of (x_t, T_t) is indicated. Note that although the three given pairs all correspond to a throughput of 53.3 kbps, they lead to very different capacity values. The reason for this is that if x_t is smaller, the initial time offset, which is constant for a fixed RTT, has to be made up for within a shorter period of time. Therefore, a higher effective throughput is required. With increasing RTT, it might not even be possible to compensate the wasted time at the beginning with higher bit rates. This is where the functions in Figure 4.10b go to infinity.



a) Principle of dimensioning with extended model
(A = 900 *kbps*, r_{peak} = 64 *kbps*)



b) Required bandwidth vs. round trip times
(A = 900 *kbps*, r_{peak} = 64 *kbps*)

Figure 4.10: Dimensioning with Extended M/G/R PS ($a_e = 900$ kbps, $r_{peak} = 64$ kbps)

4.5.2 Network Dimensioning

The use of the processor sharing formulae is the same for networks as it is for single links. However, the applicability of processor sharing in the context of networks requires some more explanation. The rationale and the dimensioning procedure are illustrated by means of a two-stage access network in Figure 4.11a. Let us assume that elastic traffic enters the network domain from the backbone and leaves it at the customer side. This would mostly be the case for web applications where considerably more traffic is transferred downstream than upstream.

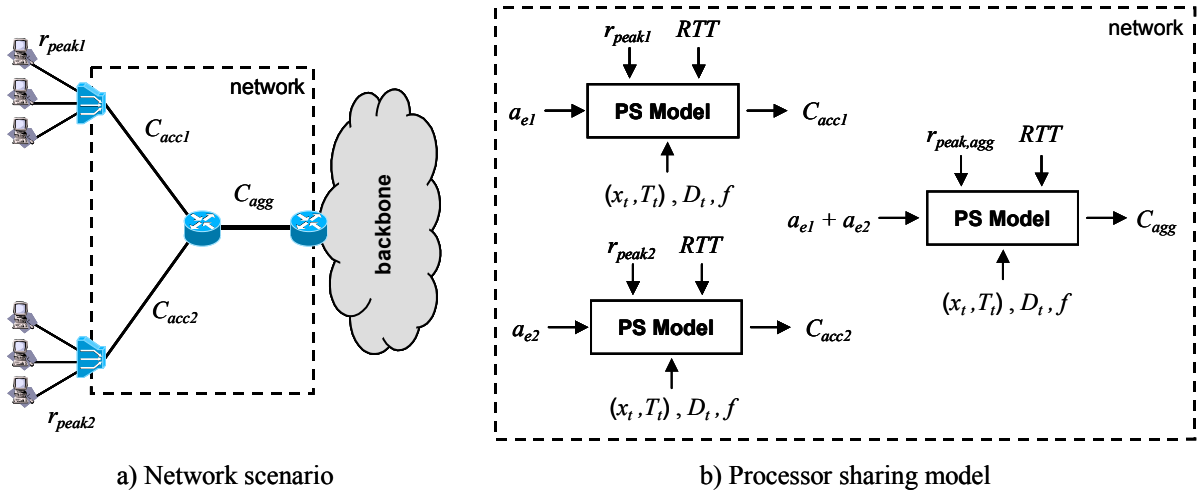


Figure 4.11: Network dimensioning with processor sharing models

The network needs to be dimensioned properly in order to fulfill the QoS requirements on an end-to-end basis. This means that the average throughput, which each flow receives from the entrance to the exit point, is D_t or that an amount of x_t bytes can be transferred within an expected time period T_t. As discussed in section 4.4.1, the rate of an elastic flow is equal along its path. Therefore, the achievable throughput of an individual flow is determined by the bottleneck link. The end-to-end delay factor corresponds to the one of the bottleneck, which is the maximum over all delay factors along the way. Noticing that throughput reductions induced by the bottleneck cannot be compensated by higher bandwidth values at other links, we might as well dimension every link in a way that it acts as the bottleneck. Therefore, the minimal bandwidths of the individual links can be calculated with the same formulae as in the single-link scenarios taking D_t, (x_t, T_t), or f as QoS criteria (see Figure 4.11b).

This procedure is an approximation as it neglects any correlations between concatenated links. However, as will be shown by following simulations, the procedure works sufficiently well for practical delay factors despite the independence assumption.

4.6 Applicability of the PS Models

In order to investigate the applicability of the processor sharing models for the dimensioning of links for TCP-based elastic traffic, numerous simulations were carried out. The networks were at first dimensioned according to the PS formulae. Then, simulation results were compared with the theoretic expectations. Our main criterion for the applicability of the models is the behavior of the average transfer time over file length. In the following sections we discuss important findings. The results suggest that the models can be applied to dimensioning.

4.6.1 Simulation Environment

We used the network simulator *ns2* [FV00] since it provides a great variety of modules and elements for IP-relevant simulations. Furthermore, due to its large user community there is considerable confidence in the validity of this simulator.

The simulator *ns2* works at packet level. However, overall we are mostly interested in the behavior at flow level, ignoring the exact packet transfers at IP and TCP layer. In our simulation scenarios we focus on the document transfer at application layer. A sender application generates files and hands them down to the transport layer. TCP then takes care of the data transfer and delivers the packets to the receiver instance. We consider a transaction to be concluded when the last packet arrives at the receiver and is handed up to the application layer. In our analysis we do not count connection establishment and termination as part of the tranfer process. Therefore, the presented transaction times do not include these extra delays.

The simulations were carried out using different TCP modules. However, as we are only interested in the TCP traffic characteristics on the rather coarse connection level, we did not observe any differences in the overall behavior. The results presented in this section were all produced with NewReno where the Delayed Acknowledgement feature was turned off.

Packet Characteristics

The packet size L_P was set to 500 bytes, which includes TCP and IP headers. During the simulations, the NewReno TCP model generates packets of equal size. Although this is not typical for general IP traffic, it reflects the fact that during file transfers, TCP instances form packets mostly of size MSS. Furthermore, the exact size of the packets is not so relevant in our simulations since we are more interested in characteristics at flow level.

Flow Characteristics

In all simulation scenarios new connections arrive according to a Poisson process, i.e., the interarrival times are negative exponentially distributed. For the amount of transferred data (including overhead) per transaction we used different distributions: heavy-tail Pareto, hyper-exponential, and negative exponential. In the following paragraphs, we will refer to the amount of data per transaction as *file lengths* or *document lengths* and use both expressions interchangeably. We did not observe characteristic differences in the outcome, which could be contributed to the file length distribution. Therefore, we will show here only results, which were generated with a hyper-exponential file length distribution. The parameters of this distribution were set to $p_1 = 0.8$, $l_{e1} = 2.5$ kbytes, $p_2 = 0.2$, $l_{e2} = 100$ kbytes, leading to an average file size of $l_e = 22$ kbytes (always including TCP and IP headers).

Dimensioning and Validation

Each scenario was dimensioned for a certain traffic load using the regular M/G/R PS model. The simulations lasted between 1000 and 5000 simulated seconds and generated at least 20000 flows. After running the simulations, the traffic load was determined. Since it might slightly differ from the preset values, the corresponding delay factors are calculated based on the simulated traffic. The applicability of the dimensioning formulae is then investigated using this delay factor. For cases with larger round trip times, where the simulation results differ from the expected values, the extended model is applied. Based on the given parameters of the simulation scenarios, the expected transfer times are calculated and then compared with the simulation results. This way, we can see if the extended model is able to better predict the real transfer times.

Other Remarks

The TCP modules of *ns2* reflect the functionalities of realistic implementations on a very detailed level. This allows making concrete statements about the behavior and the performance of the available protocols. On the downside, it makes it also difficult to simulate network scenarios with many nodes and high traffic loads. Therefore, our simulations focus on the characteristics of packet flows over links with lower bandwidths. Nevertheless, this does not limit the significance of the results. For large-bandwidth links with highly aggregated traffic flows, the delay factor is quite robust against traffic variations.

4.6.2 Single Bottleneck Simulations

Single-link scenarios are used to show the fundamental applicability of the processor sharing models. Based on these cases, important effects concerning the dimensioning for elastic traffic can be demonstrated. These effects also arise in similar ways for network scenarios with several links. For all simulations with a single bottleneck we use the topology shown in Figure 4.12.

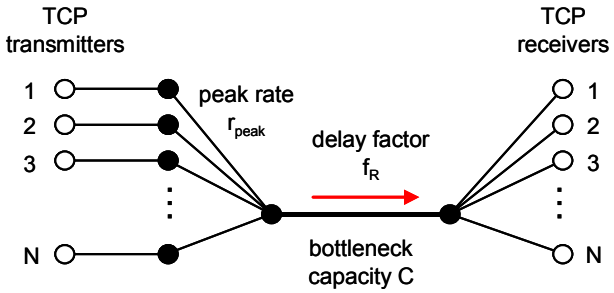


Figure 4.12: Single-bottleneck simulation scenario

Applications on the left hand side generate files and hand them down to the TCP layer. A TCP sender transmits the data to its respective counterpart on the right hand side. On its way to the destination, every traffic flow traverses the bottleneck link with capacity C. The one-way propagation delay of the bottleneck link is T_P^b and its buffer size in packets is B. Every sender node is connected to the bottleneck link via an access link with bandwidth r_{peak} and propagation delay T_P^a. In all simulations, the receiver window wnd was set large enough in order to not introduce any limitations.

In total, a pool of N TCP sources is set up, which are activated by a central simulation manager. The number N of potential TCP sources is set large enough, so a Poisson flow arrival process can be generated. For our simulations, N was between 200 and 500. By checking the maximum number of active flows at the end of a simulation, we made sure, that N did not introduce any limitations, which would disturb the outcome.

4.6.2.1 Reference Scenario

This scenario serves as the reference for further investigations. The link bandwidth is chosen in a way to obtain a delay factor of around 1.2. Furthermore, the overall propagation delay is mainly determined by the bottleneck link and is therefore equal for all flows. In summary, we have following network parameters and simulated traffic characteristics:

- incoming traffic: $\lambda_e = 20$ 1/s, $l_e = 22.22$ kbytes, $a_e = 3555$ kbps
- access links: $r_{peak} = 64$ kbps, $T_P^a = 50$ μs
- bottleneck link: $C = 3712$ kbps, $T_P^b = 10$ ms, $B = 500$
- expected delay factor $f_R = 1.27$

The graph in Figure 4.13 shows the average transfer times (and their 95% confidence intervals) $T(x)$ for different file sizes x. Every point $(x, T(x))$ of the curve represents an average time value t taken over a larger number of file transactions, for which the file size was roughly equal to x. In addition to the simulated mean values, the theoretical transaction times predicted by the M/G/R PS model are given.

In the remainder of the chapter, this type of graph will be the preferred way to present simulation results and to investigate the quality of the theoretical prediction. If not mentioned otherwise, the theoretical values refer to the pure M/G/R PS model without any extensions.

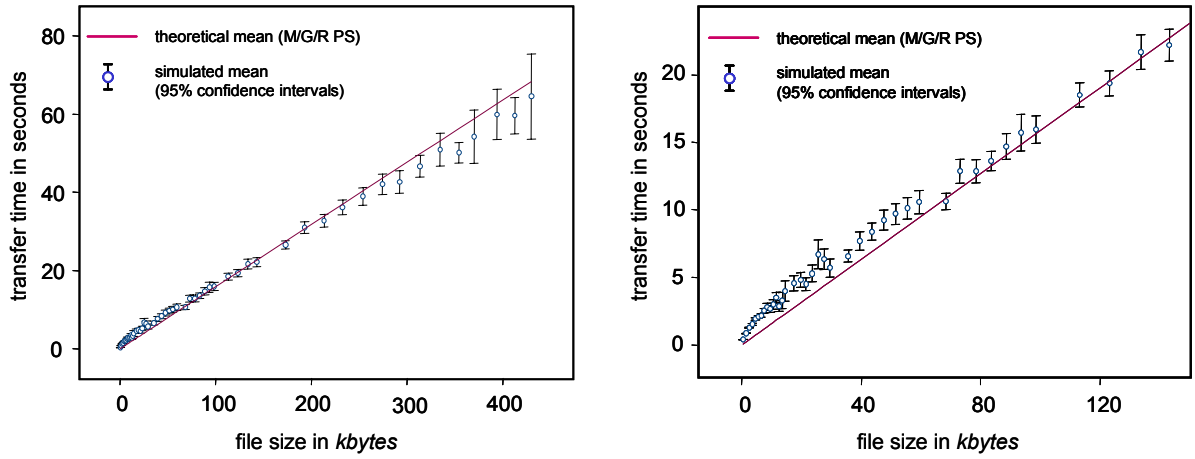


Figure 4.13: Transfer times vs. file size (constant delay factor)

As we can see from the graph, the M/G/R PS model closely predicts the average behavior of the TCP transactions. However, smaller files are somewhat disadvantaged. This can clearly be seen in the graph on the right hand side in Figure 4.13, which zooms in at smaller file sizes (note the different scale of the axes). Small files experience a somewhat larger average transaction time than expected. While the difference between the simulated and the expected values can be quite high in relative terms (up to a factor of about 4), the absolute increase is rather small. This is where we argue that a user does not mind or even notice the proportionally longer transfers of little files as long as larger ones receive their expected throughput.

The mean duration of longer TCP transactions is mostly even shorter than expected. The explanation for this is that large files have a long enough transaction time for them to see the average behavior of the system and to make up for the lost time during the slow start phase. As newly arriving flows first have to gradually build up their send rate, existing flows are not forced to reduce their current bandwidth shares right away. This slow-down is intensified by the fact that queuing delay at the bottleneck link contributes to the overall round trip time.

Therefore, if a new flow enters the system at a time when a lot of other flows are active, it goes through a longer start up phase due to the higher buffer occupancy at the bottleneck. For short flows the duration of this phase is proportionally longer than for long ones.

Small files are slightly disadvantaged and exceed their delay requirement, while long file transactions are able to take advantage of this short coming. Speaking in terms of network dimensioning, this would mean that the theoretical formula underestimates the capacity needs for short flows, which might lead to unwanted quality degradation. However, for increasing file sizes the theoretical model becomes more accurate and, finally, even overestimates the transaction time. Thus, for larger file transactions dimensioning according to the processor sharing models gives results, which are quite accurate and usually on the safe side. The same effect has been shown for all types of simulations.

So far, the graphs have shown only the average of the transfer times over many transactions. One has to be aware that this does not mean that all transactions are finished at times close to their expected mean. This is illustrated in Figure 4.14. The graph shows a box plot of normalized transaction times for different file sizes. From all transactions, we extract samples with approximately the sizes of 5, 10, 20, 50, 100, and 150 kbytes. The transaction times are normalized with respect to the M/G/R PS values e.g., for 100 kbytes this would be 15.9 seconds. Thus, a value of 1 corresponds to the expected value, while values greater than 1 indicate longer than expected transactions. Each box summarizes the statistical characteristics of the transactions. The white line indicates the median, whereas the box itself marks the middle half of the data. The whiskers indicate the samples, which are the farthest from the quartiles but not beyond 1.5 times the inter-quartile range. Additionally, outliers are shown. As can be seen (and also has been observed earlier), small files are mostly above their expected mean. The median for files of size 5 kbytes is at 2.8. Also, the relative spread of the small files is larger, having outliers, which show a factor of up to 10. However, for most transfers the extra delay is not considered to be too critical, since the normalizing constant, which is the expected transfer time, is only 0.8 s. From the distribution of transfer times (not shown here), we observe that the great majority of transactions (about 96%) is finished within 4 seconds (factor 5). For increasing file sizes, where strong deviations would become more noticeable and also more critical, we see that the bulk of transfer times is centered around the median, and also getting closer or even falling below the value of one. Since absolute transaction times are longer for large file sizes, this effect is very positive.

Overall we conclude that the model works fine for this scenario. In the following scenarios, we look at corresponding graphs. For each of them, a dimensioning model seems applicable as long as the sojourn time behavior of the simulations follows approximately the theoretical expectations. Slight differences cannot be avoided and have to be accepted.

4.6.2.2 Long Propagation Delays

For this simulation we increase the propagation delay of the bottleneck link to 250 ms. From publications presented in Section 4.3.1 we know that long round trip times negatively affect the performance of TCP and also its rate sharing mechanisms. Therefore, we want to find out, how the increase affects the transfer times of files and, therefore, the accuracy of the dimensioning formulae. We have following parameter values:

- incoming traffic: $\lambda_e = 20$ 1/s, $l_e = 21.8$ kbytes, $a_e = 3496$ kbps
- access links: $r_{peak} = 64$ kbps, $T_P^a = 50$ μs

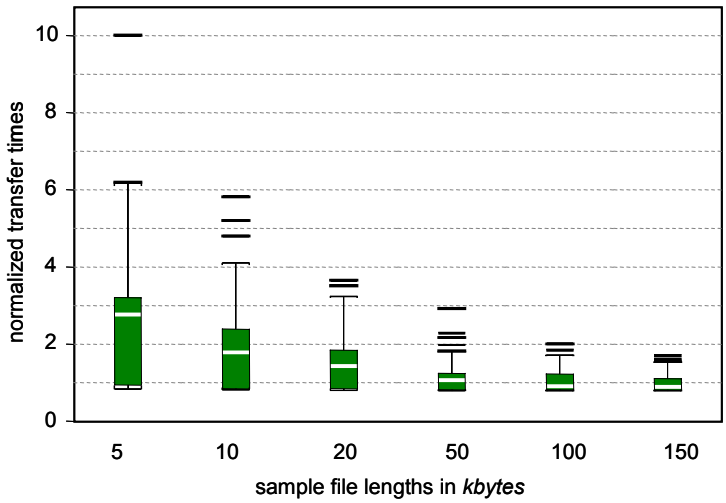


Figure 4.14: Box plot of transfer times for reference scenario

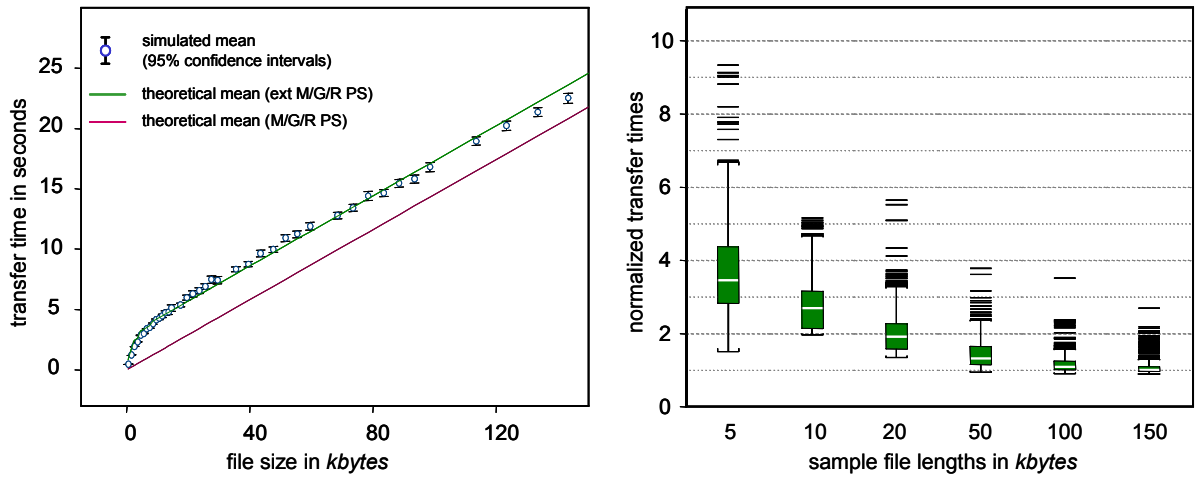
- bottleneck link: $C = 3712$ kbps, $T_P^b = 250$ ms, $B = 500$
- expected delay factor $f_R = 1.16$

This scenario is similar to the one before, only that now the propagation delay on the bottleneck link is 250 ms instead of 10 ms. The first observation we make is that now the mean transfer time curve in Figure 4.15a does not cross the theoretical line of M/G/R PS as it did in the case of 10 ms propagation delay. Although the slope is equal, an offset exists, which lets the simulated data lie above the expected ones. It is obvious that the regular M/G/R PS model has problems predicting the transfer time for small files when the round-trip time increases. Our extended model, which takes into account the slow start mechanism, in this case gives more accurate results. The curve in the graph was computed by making use of an estimated round trip time, which also includes queuing delay in the buffer of the bottleneck link. We assume that in times where more than R flows are active (i.e., the sum of the peak rates exceeds the link capacity C), the buffer fills up quickly and the queue is mostly at its limit. Thus, in addition to its own service time, each packet has to first wait until all other packets in the buffer have been served. This corresponds to serving B packets of length L_P with capacity C. In times of $n < R$ we assume the queue is approximately empty. With this we estimate the average round trip time as follows:

$$RTT_{est} = P(n \leq R) \cdot \left(2T_P^b\right) + P(N > R) \cdot \left(2T_P^b + \frac{B \cdot L_P}{C}\right)$$

The state probabilities $P(n)$ of having n active jobs in the system are given in [Coh79]. Thus, we can determine the probabilities $P(n \leq R)$ and $P(n > R)$. In the given scenario, we have $P(n > 58) \simeq 0.5$ resulting in an estimated average round trip time of $RTT_{est} = 769$ ms.

Furthermore, we are again interested in the spread of the transaction times for different transaction volumes (Figure 4.15b). The effects are very similar to the reference scenario. For small transactions, the variation is rather high, resulting in a median level that corresponds to about 3 times the theoretical mean value. For increasing file sizes, however, the relative variation is reduced. The box plot illustrates these observations.



a) Transfer times vs. file size

b) Box plot of transfer times

Figure 4.15: Simulation results for scenario with long round trip time

With increasing propagation delay these effects become more pronounced. In Figure 4.16 the transfer time is illustrated for various large propagations delay ($T_P = 100$ ms, $T_P = 250$ ms, and $T_P = 500$ ms). The transfer-time graphs show the typical offset and for longer files converge to the linear asymptote whose slope corresponds to the delay factor.

4.6.2.3 Low Capacity

With this scenario we investigate the accuracy of the formulae for links with smaller capacities and similar utilization level. For this constellation the delay factor is very sensitive to load variations (see Figure 4.4). This adds some uncertainties to the dimensioning process, as it is always difficult to exactly predict the offered load. Therefore, we want to find out if TCP contributes even more to the uncertainty or whether it works as approximated by the processor sharing formulae. The traffic load and the bottleneck capacity are both reduced in a way to give a link utilization of 0.96. The simulated parameters are:

- incoming traffic: $\lambda_e = 10$ 1/s, $l_e = 21.9$ kbytes, $a_e = 1746$ kbps
- access links: $r_{peak} = 64$ kbps, $T_P^a = 50$ μs
- bottleneck link: $C = 1800$ kbps, $T_P^b = 10$ ms, $B = 500$
- expected delay factor $f_R = 1.98$

Figure 4.17 shows the sojourn time graph as well as the box plot for this scenario. It is obvious that the M/G/R PS model again fits quite well. The negative effect of underestimating the transfer time is limited to small file transactions. While small transactions are considerably longer than expected (in relative terms!), large files again take advantage and achieve shorter transaction times. The low capacity leads to longer queuing delays in the buffer at the bottleneck. A full buffer corresponds now to an extra delay of 1.1 s, which has the same effect as long round trip times. It slows down the start up phase of TCP connections

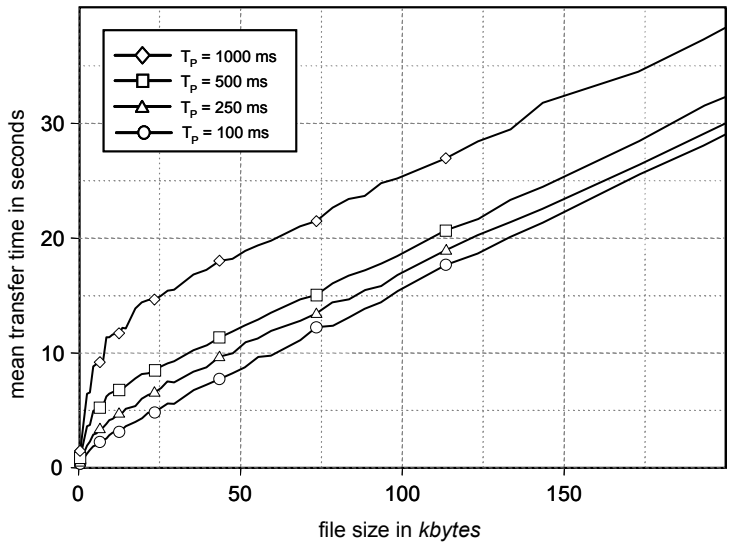


Figure 4.16: Effects of long round trip times

leading to throughput degradation of short flows. Following the method presented in the preceding scenario, the round trip time can again be estimated. The corresponding parameters are: $R = 27$, $P(R > 27) \simeq 0.8$, and $RTT_{est} = 0.89$ ms. As we can see in Figure 4.17a the Extended M/G/R PS model is capable of accounting for the increase of RTT due to high buffer occupancy.

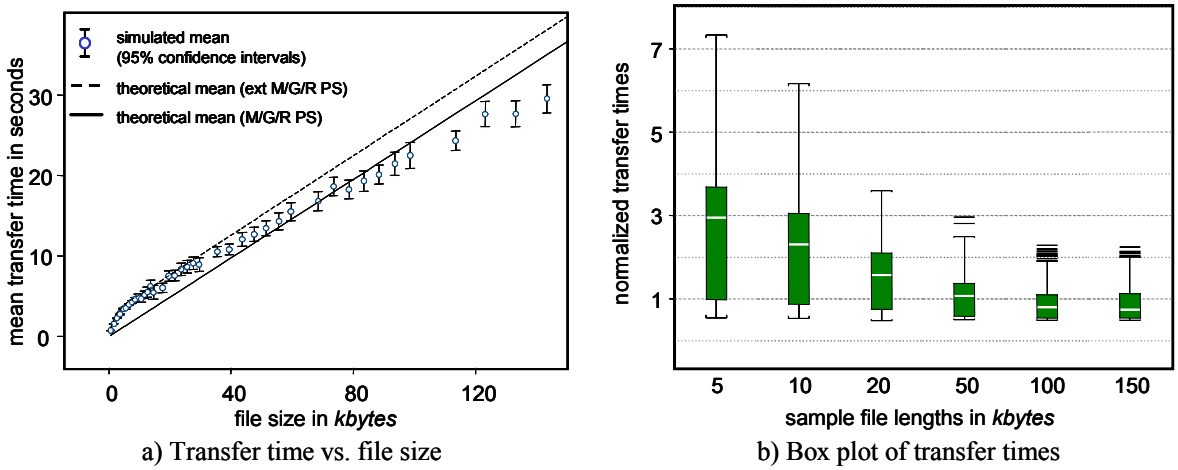


Figure 4.17: Transfer time characteristics for low-capacity scenario

4.6.2.4 Different Delays

In this simulation scenario a heterogeneous environment is investigated with two groups of TCP connections that experience different round trip delays: one group with small propagation delays of 10 ms and one group, whose access links introduce large delays of 100 ms.

In practice, the large delay is not necessarily introduced by the access links but by paths through backbone networks. However, as TCP regulates traffic in an end-to-end manner, the extra delay has the same effects, irrespective of where it is induced. The key question of this scenario is whether TCP connections with small delays can exploit bandwidth-sharing deficiencies of the flows with longer round trip times. We have following parameters:

- incoming traffic type 1: $\lambda_{e1} = 10$ 1/s, $l_{e1} = 22.3$ kbytes, $a_{e1} = 1785$ kbps
- incoming traffic type 2: $\lambda_{e2} = 10$ 1/s, $l_{e2} = 22.17$ kbytes, $a_{e2} = 1774$ kbps
- access links type 1: $r_{peak} = 64$ kbps, $T_P^{a1} = 10$ ms
- access links type 2: $r_{peak} = 64$ kbps, $T_P^{a2} = 100$ ms
- bottleneck link: $a_e = 3559$ kbps, $C = 3712$ kbps, $T_P^b = 1$ ms, $B = 500$
- expected delay factor $f_R = 1.28$

The outcome, which is illustrated in Figure 4.18, is consistent with earlier findings in two aspects. First of all, a higher propagation delay (here 100 ms) leads to prolonged start up phases, which again cause proportionally longer transaction times for smaller flows. This has already been observed in the homogeneous simulation environment with a propagation delay of 250 ms. The second observation concerns the performance of long transactions, which again take advantage of the short flows' inability to utilize their share. For both round trip time classes, long flows cross the theoretical line. However, we see that connections with short round trip times outperform their counterparts with higher propagation delay.

Nevertheless, from a dimensioning perspective, the processor sharing model seems to provide sufficiently exact predictions for the elastic traffic flows.

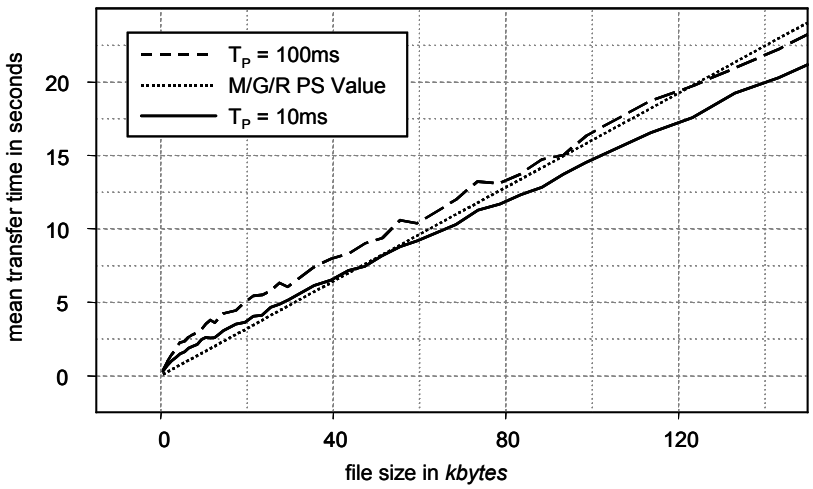


Figure 4.18: Transfer times vs. file size for scenario with different delays

4.6.2.5 Different Peak Rates

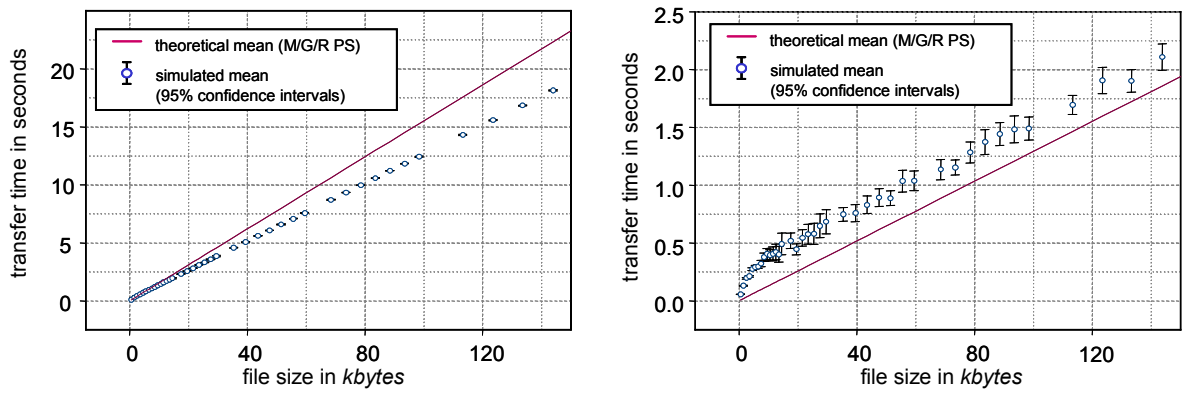
Similar to the preceding investigations, we again look at a heterogeneous environment. This time we introduce two types of access links, one with a peak rate of 64 kbps (ISDN) and the other one with 768 kbps (typical x-DSL). This scenario is very interesting as it mixes two classes of customers, who might have different expectations about their quality of service. However, only one notion of QoS can be offered since all traffic is carried in the same service class.

As discussed in Section 4.5, mixing flow classes with different peak rates introduces a problem when calculating the delay factor or – from the perspective of dimensioning – when calculating the required capacity. As there are no exact formulae, which consider multiple peak rates, an approximation is required. We use Lindberger's approach and take the weighted average of the two peak rates, which in the given scenario is 413 kbps. With this peak rate and the parameters listed below, the delay factor should be about 1.24. The capacity of the bottleneck link in this case is 4412 kbps. If all flows had a peak rate of 64 kbps, a capacity of 3766 kbps would suffice to achieve the same delay factor. For traffic with exclusively 768 kbps peak rate, the same calculation would give a capacity of 4864 kbps.

- incoming traffic 1: $\lambda_e = 10$ 1/s, $l_e = 22.54$ kbytes, $a_{e1} = 1811$ kbps
- incoming traffic 2: $\lambda_e = 10$ 1/s, $l_e = 22.4$ kbytes, $a_{e1} = 1785$ kbps
- access links 1: $r_{peak} = 64$ kbps, $T_P^{a1} = 50$ μs
- access links 2: $r_{peak} = 768$ kbps, $T_P^{a2} = 50$ μs
- bottleneck link: $a_e = 3596$ kbps, $C = 4412$ kbps, $T_P^b = 10$ ms, $B = 500$
- expected delay factor $f_R = 1.24$

The simulation shows that the flows with the small peak rates achieve shorter transaction times than expected (Figure 4.19a). From the graph we can actually determine an effective throughput of almost 64 kbps, the flows' peak rate. Thus, the simulated delay factor for the low-peak-rate flows is almost 1 and the QoS is a lot better than desired.

The high-peak-rate flows experience a somewhat lower QoS than desired. This degradation is partly due to the round trip times, which - in relative terms - contribute a considerable amount of time to the overall transaction times. Another reason for this unfairness is that whenever a high-peak-rate flow enters the system, it greatly increases the probability that the system is in a rate-sharing state (alone through its own presence). Since low-peak-rate flows require only small bandwidth shares, they do not cause the system to change into a rate-sharing state that easily. This effect is in principle similar to the unfairness in pure multi-service loss systems, where different connection types have different bit rates. This bias against larger peak rates has also been pointed out by Lindberger for Available Bit Rate (ABR) traffic in ATM networks [Lin98]. However, note that the transaction times for high-peak-rate flows are still an order smaller than the transaction times of low-peak-rate flows. Only since the delay factor is a relative QoS measure the actual QoS is considered to be worse.



a) Transfer time of flows with $r_{peak} = 64$ *kbps*

b) Transfer time of flows with $r_{peak} = 768$ *kbps*

Figure 4.19: Transfer times for scenario with different peak rates

4.6.2.6 Large Peak Rates

Finally, we present a scenario with large enough peak rates so that each flow can utilize the bottleneck link by itself. This scenario is used to demonstrate the applicability of the M/G/1 model. Since transaction times should be in the order of the preceding scenarios, we choose a delay factor of about 25. The parameters and simulated values are set as follows:

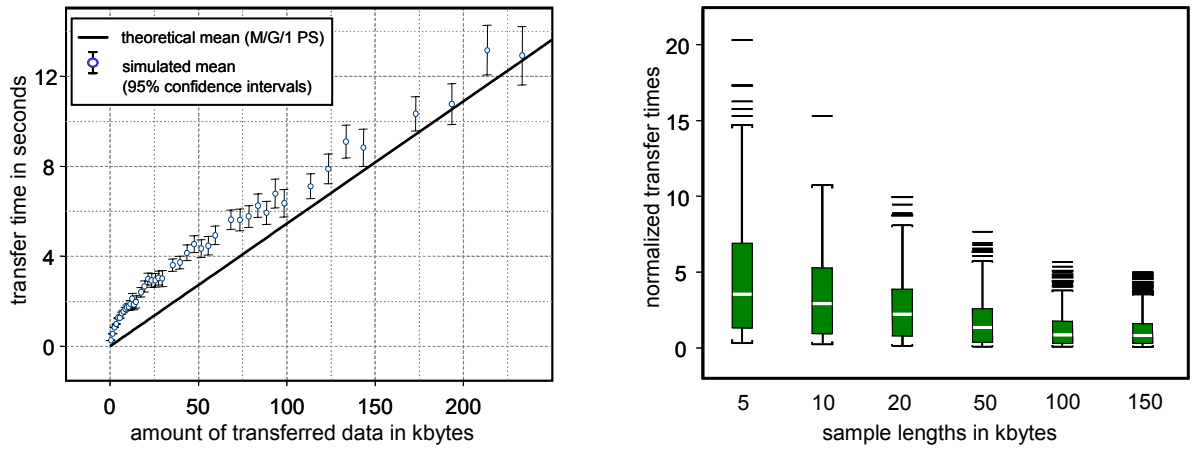
- incoming traffic 1: $\lambda_e = 20$ 1/s, $l_e = 22.2$ kbytes, $a_{e1} = 3565$ kbps
- access links 1: $r_{peak} = 10$ Mbps, $T_P^a = 50$ μs
- bottleneck link: $C = 3712$ kbps, $T_P^b = 10$ ms, $B = 500$
- expected delay factor $f_R = 25.25$

Both graphs of Figure 4.20 – transaction times over file size as well as the box plot of the normalized transaction times – suggest that the M/G/1 PS model is indeed capable of predicting the achievable QoS. We have the same asymptotic behavior of the transfer time as with all M/G/R PS cases. Small files cannot achieve the theoretic values, while larger files come close to them (Figure 4.20a). Also, the spread of the transaction times around their median is higher for short transfers than it is for large ones (Figure 4.20b).

4.6.3 Tree-structured Access Networks

After presenting simulations with a single bottleneck link, we now investigate a typical tree-structured IP network as it is often used in the access area of larger networks or also in typical enterprise networks. However, it should be noted that the physical structure does not have to be a tree form. We consider here the logical structure as it appears to IP. The main characteristic of the tree-structured access network is that traffic is aggregated in several stages (in our case, there are two aggregation stages). The backbone section of the network is modeled by means of a single node.

As discussed above already, the focus is on access networks as they are most delicate for network dimensioning. On the one side, great savings can be achieved in the access areas. On



a) Transfer time over amount of transferred data

b) Box plot of transfer times

Figure 4.20: Scenario with large peak rates

the other side, the delay factor is very sensitive to traffic variations for small capacities. Thus, especially in the access area a thorough dimensioning procedure seems to be most beneficial.

Figure 4.21 depicts the simulation setup. A total number of 1000 TCP transmitters is assigned to 10 access nodes evenly. The ten access nodes are connected to the aggregation node from where a link is established to the backbone node. The TCP receivers are directly connected to the backbone node. Every TCP source corresponds to a TCP sink. Files are generated at the sources and are sent to the respective sinks. All links are full duplex with equal capacity in both directions.

We have also carried out the simulations for the same topology only with the location of the transmitters and receivers exchanged. This scenario would then correspond to an access network where customers download information from the world wide web or from proxy servers within the IP network. The results were the same.

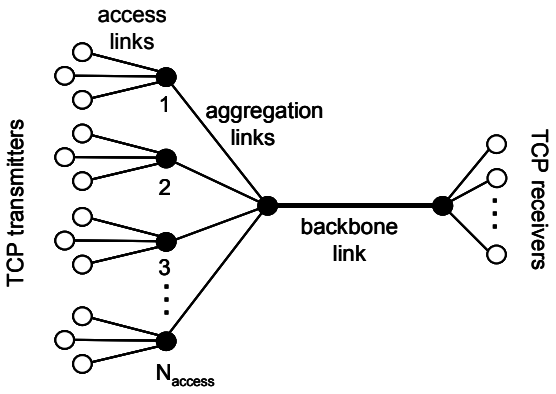


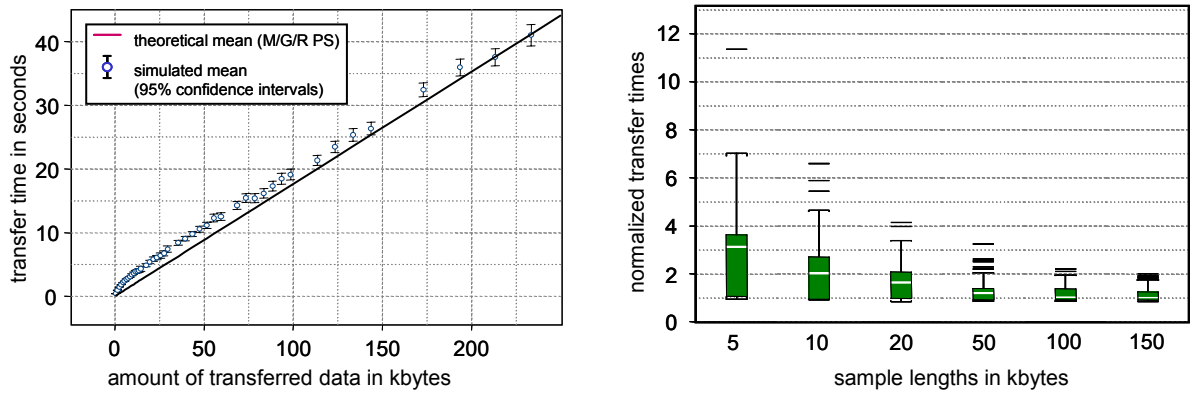
Figure 4.21: Simulation scenario for access networks

For all access network scenarios that we have simulated we observed analog effects as in the

cases of the corresponding single-link scenario. Therefore we only present one example, which demonstrates the applicability of the proposed dimensioning procedure for tree-structured IP networks. It is a homogeneous environment where all sources have the same peak rates. In each area the same amount of traffic $a_{e,i}$ is generated, with the same flow arrival rate $\lambda_{e,i}$ and file lengths l_e. The simulated values are as follows:

- access links: $r_{peak} = 64$ kbps, $T_P^{acc} = 50$ μs
- aggregation links: $C = 477$ kbps, $T_P^{agg} = 50$ μs, $B_{agg} = 500$
- backbone link: $C = 3712$ kbps, $T_P^{bb} = 10$ ms, $B_{bb} = 500$
- expected delay factor: $f_R = 1.41$

As indicated in Figure 4.22, the results for the access network are very similar to the ones for the single-link reference scenario. For smaller file transfers we have again the extra delay due to slow start, buffer delay, and round trip times, while longer files are able to achieve the desired QoS. The same is true for the relative spread of the transaction times. Small files deviate considerably from the expected value. With increasing file size the variation is reduced.



a) Transfer time over amount of transferred data

b) Box plot of transfer times

Figure 4.22: Simulation results for access network

4.7 Summary

In this chapter the capacity assignment process for elastic traffic was addressed. We presented dimensioning models, which are based on the theory of processor sharing, and demonstrated their applicability through simulations. Overall, the simulation results comply with the theoretical expectations sufficiently well. This suggests that the processor sharing formulae can indeed be used for dimensioning links and networks for elastic traffic. However, for scenarios with longer round trip times, the basic processor sharing model shows some deficiencies. It overestimates the average performance of the individual flows and, as a consequence, leads to underdimensioning. Therefore, we propose an extension, mainly for the consideration of

TCP's slow start mechanism, which significantly improves the prediction accuracy and, thus, the determination of the appropriate capacities.

Chapter 5

Dimensioning for Stream Traffic

In this chapter the dimensioning process for stream traffic is investigated. As such we subsume all traffic, which is generated by real-time applications with strict QoS requirements. Typical services in this category comprise telephony or audio and video conferencing. Analog to the preceding chapter, we consider traffic characterization at flow level. Each flow corresponds to a connection or a call, which is established between a sender and a receiver and which requires a certain bandwidth in order to achieve the desired quality of service. In this chapter, we will use the expressions "flow", "connection", and "call" interchangeably. Each one denotes a unidirectional flow of packets between two specific end points, which are using the considered service.

We focus on optimization issues, which arise from different realizations of call admission control (CAC) schemes. As CAC concepts for IP networks are currently still a matter of research, we analyze various possibilities and investigate their consequences for network dimensioning. Based on an abstracted view of the schemes, we discuss the overall optimization problem and derive appropriate solution approaches. Due to the complexity of the problem it cannot be solved as a whole. Therefore, we present a two-step methodology for computing the capacities or bandwidth shares [RBF03]. Finally, the methodology is applied to our sample scenarios.

5.1 Overview and Objectives

The dimensioning process considers individual network domains, which connect customers through access networks and other domains through peering points. The origins and destinations of stream traffic can lie within the current domain (in an attached access area) or somewhere outside in other domains. Whenever connections need to be established within or across domains, appropriate intra- and inter-domain call signaling protocols are required. However, currently no such protocols are standardized, yet. Therefore, we will briefly describe existing approaches when presenting the abstract CAC models.

The nodes of a network are usually divided into classes of core (or internal) nodes and border nodes. The latter denote the points, at which traffic enters and leaves the network, while the former only forward traffic within the network. From MPLS we adopt the expressions *ingress* and *egress* nodes, without requesting that MPLS is actually applied in the network. Furthermore, we extend the notion of ingress and egress to links. An ingress link is a link, which establishes the connection from a peering partner or from the access network

to the considered network domain. The egress link is the corresponding link only in opposite direction (since we are considering unidirectional links). In our procedure we assume the most general case, where each router implements ingress as well as egress functionalities and also acts as transit node. Figure 5.1 illustrates the network model.

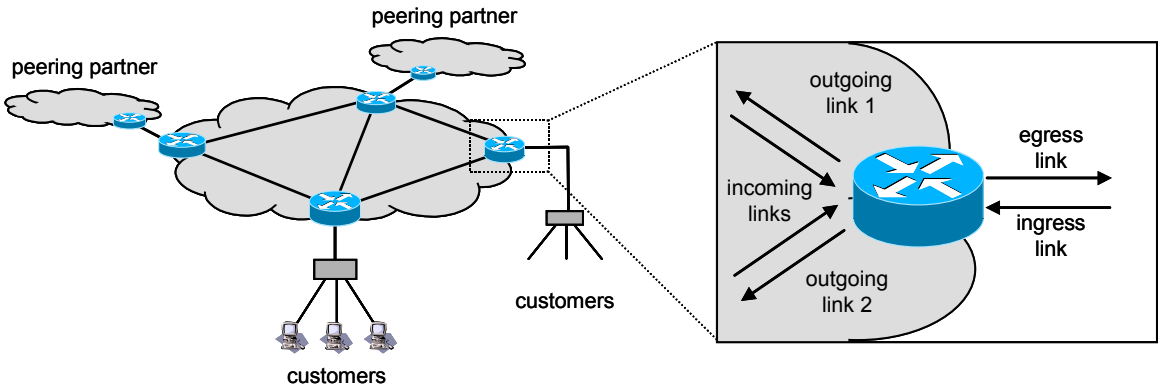


Figure 5.1: Network domain for stream traffic

As we look at one type of service and one customer group per dimensioning process, we have only one ingress link per ingress node. If several customers with different QoS requirements are connected, a customer-specific CAC would have to be applied so that no customer could take away another one's bandwidth. Therefore, they would have to be treated independently from each other (and so would be the network dimensioning process). However, if the service is offered to several customers who request the same QoS, we can aggregate their traffic. Then, it is again sufficient to consider only one incoming link per ingress router. We also assume that there is only one egress link per router, which corresponds to one neighbor domain per egress point. Since the dimensioning procedure, which takes into account multiple egress links, is analog to the procedure for single egress links, we omit this case during the following discussions. At the end of the chapter, the necessary modifications are pointed out.

The objective of the dimensioning process is to minimize the capacities or bandwidth shares, which are assigned to real time traffic in the network. Doing so, it has to be assured that different QoS requirements are met. Following important issues motivate our dimensioning problem and define the cornerstones of the optimization procedure:

- A stream connection, which is established between a sender and a receiver, requires a certain bandwidth on each link along the path in order to function correctly with the desired packet-level QoS. Failure of providing this bandwidth on any of the links (even temporarily) causes service degradation or interruption due to packet loss or delay.

- Users of real time services do not accept service degradation or interruption once they have started a session. They would rather have the connection to be blocked whenever the network is not able to carry it with the appropriate QoS. However, the blocking probability should not exceed a certain threshold, as this would again degrade the experienced service.

As a consequence, it is assumed that a notion of *connection* exists in the network. Each packet flow can be assigned unambiguously to an active connection. Furthermore, it is as-

sumed that it is possible for each considered application to specify a bandwidth value, which is required to assure appropriate QoS. This bandwidth value is often referred to as effective bandwidth. It describes the interrelation between the traffic characteristics and relevant QoS parameters. As will be explained later in this chapter, we require additive bandwidth values. This is not always the case for effective bandwidth formulae. However, some popular applications such as Voice over IP produce traffic streams with almost constant bit rates (see for example [SRGT03]). In these cases, the bit rates are additive. Another context where linearly additive bandwidths are a valid assumption is worst-case dimensioning as proposed in the IntServ framework. We only consider homogeneous traffic flows within one traffic class, which require the same bandwidth. Thus, when referring to the bandwidth demand of a traffic aggregate, we can either specify the total bandwidth demand or just the number of active calls. Since we know the required bit rate of an individual call, we can easily transform the one into the other.

The traffic at connection level is either specified through parameters *connection/flow arrival rate* λ_s and *mean holding time* l_s or the value of the respective traffic volume in Erlang (which is the product of connection arrival rate and holding time). Interarrival times and holding times are assumed to be exponentially distributed.

Since service degradation and interruption should be avoided, a connection has to be protected throughout its life time. To do so, one has to be able to detect the start and also the end of a connection. Furthermore, some form of call admission control needs to be implemented, which regulates the access of flows into the network. Only then it is possible to assure that established calls are not disturbed by newly arriving ones. The exact way that CAC should be realized in IP networks is still an open issue. Various concepts exist, which in an abstract way will be presented in the next section.

In summary we can loosely state our optimization problem as follows:

- We would like to minimize the capacity shares in the network, which need to be reserved for stream traffic,
- while keeping the call blocking probability below a certain threshold and
- providing sufficient bandwidth to active calls throughout their life time.

A more detailed discussion of the problem and the outline of the solution procedure first requires the presentation of the considered CAC schemes and their abstraction. They provide the basis for the exact formulation of the optimization problem.

5.2 Call Admission Control Schemes

Many CAC concepts have been proposed over the past years, their complexity ranging from very simple to very complex. In order to get a grip on the different CAC schemes (so we are able to dimension the network appropriately) we discuss the different possibilities of implementing call admission control in IP networks. It is not our intention to explain the individual components and functionalities of CAC architectures or to provide an exhaustive overview of proposed schemes. We focus mainly on aspects, which are relevant to network planning, and derive an abstract CAC functionality model, which is able to capture the key effects.

The crucial aspects of any CAC scheme from the perspective of network dimensioning, is the location of the bandwidth components, which are subject to call admission control. As we will see in a moment, it is not necessarily the case that all links along a flow's path are checked as it is done in circuit-switched networks. There are possibilities, which only perform CAC at dedicated links in the network. For admission control at a link, a certain configured bandwidth value is taken into account. This could either be the total capacity of the link, or just a certain share of the overall capacity. To clarify this notion of bandwidth, we introduce the expression *CA-controlled bandwidth*. The CA-controlled bandwidth, which is part of the reserved, class-specific bandwidth, provides the basis for a CAC instance to make its decision. Overall, we have following picture: from a link with capacity C a certain share $BW \leq C$ is reserved for stream traffic of a certain type (e.g., by means of weighted fair queuing). Part or all of BW, namely $BW_{CAC} \leq BW$, might be subject to CAC. A controller takes into account this bandwidth BW_{CAC} when making an admission decision, which affects the link.

To illustrate the following discussion of the individual CAC schemes, we take the network scenario shown in Figure 5.2 as an example. The traffic flows enter the network at nodes A and B and are destined to nodes C, D, and E. The routes between ingress node A and all egress nodes are indicated in the figure. For calls, which enter the network through node B, the routes are chosen accordingly. For simplification we assume that it is sufficient to provide bandwidth for only one connection between each ingress node and every egress node. Note that we consider flows, which traverse our network from the left (from ingress routers) to the right (to egress routers). However, this is done only for illustrational purposes. As we have pointed out earlier, each node could implement all functionalities of an ingress, an egress, as well as a transit node.

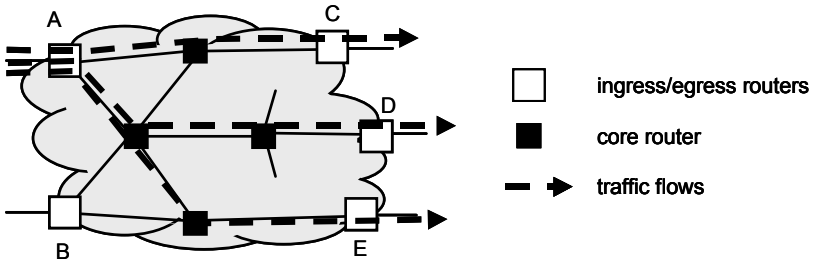


Figure 5.2: Network scenario with flows between ingress and egress routers

5.2.1 Call Admission Control at Ingress Nodes

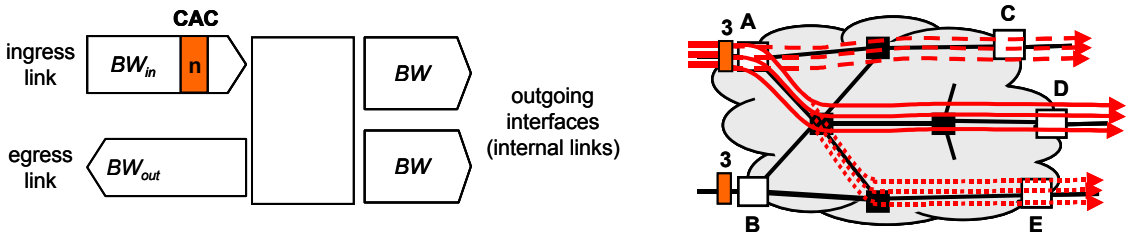
A simple method to avoid overload within the network is to perform call admission control only at the ingress nodes and to provide sufficient capacities within the network. When making the admission decision, a control entity at the ingress router only needs to know about locally available information. This has the advantage that the control unit can act independently of any other controller in the network. A global view of the whole network as well as information exchange between distributed control entities are not required. However, as we will see later on, the downside of this approach is the increased demand for bandwidth necessary to assure proper QoS.

Basically three variations of ingress CAC exist, which differ in the bandwidth component, which is considered for admission decisions:

- CAC per ingress link
- CAC per outgoing interface of the ingress router
- CAC per destination.

5.2.1.1 CAC per Ingress Link (*I-CAC*)

CAC per ingress link (*I-CAC*) considers the available bandwidth at the ingress link and the number of already active calls for admission decisions. Thus, the CA-controlled bandwidth in this case is equal to the total reserved bandwidth of the ingress link. In Figure 5.3a the functionality of this type of CAC is illustrated. All calls arriving at the same ingress router are subject to the same CAC instance, irrespective of their destination. The number n in the shaded box of the ingress link arrow symbolizes the maximum number of calls, which are admitted by the controller.



a) Node model for *I-CAC* b) Possible flow distribution

Figure 5.3: CAC at ingress per ingress link (*I*-CAC)

Possible Realization

This CAC scheme can be realized by placing control entities at every ingress node and by configuring them with the maximum number of incoming calls. A concrete realization for an *I-CAC* scheme in a single domain would be an H.323 system with the gatekeeper-routed call signaling option [ITU00a, ITU00b, KKS01]. In this case the call setup messages between user terminals are routed through the respective gatekeepers, which have to authorize every outgoing call. Thus, a gatekeeper could be used to regulate the number of calls entering the network domain from its assigned access area. This CAC realization, however, only works for the originating domain since gatekeepers do not have knowledge about routing and, therefore, do not contact gatekeepers on the way. If this ingress-based CAC scheme should be implemented on a domain hop-by-hop basis, an appropriate backend service with an interface to the gatekeeper would have to be developed.

Consequences for Network Dimensioning

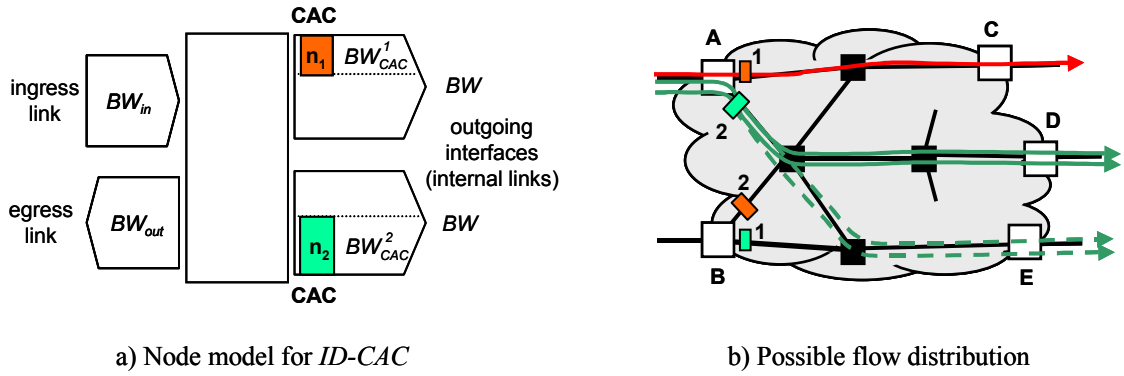
A network designer has to be aware that the *I-CAC* scheme limits only the traffic inflow at each ingress node but not the distribution of the traffic within the network. The acceptance

decision for a call is made irrespective to its destination. Thus, situations such as the one illustrated in Figure 5.3b could arise. The ingress node A allows three concurrent calls into the network, indicated by the number above the CAC box. Due to CAC at the ingress link all three flows could be destined to the same egress node. Taking into account another three calls coming into the network through ingress node B, a total of six connections could go to each of the egress nodes simultaneously.

In order to avoid service degradation at packet level at all times, the network has to be robust against all possible traffic variations. This can only be achieved if the dimensioning process considers the whole range of possible flow patterns and all links (also the egress links) are dimensioned for the maximum amount of traffic, which they might have to carry.

5.2.1.2 CAC per Router Interface (*II-CAC*)

A somewhat stricter strategy of resource control at the ingress node is interface-specific CAC (*II-CAC*). Instead of having one CA-controlled bandwidth component for all incoming calls, the admission decision is now done on a per-outgoing-interface basis. This way, a router first determines the outgoing direction of a new call and then decides, whether it can still fit it onto the respective link. Along the path towards the destination, no further call admission control is carried out in this domain. The corresponding ingress node architecture is depicted in Figure 5.4a. The CAC decision for each flow is based on the CA-controlled bandwidth BW_{CAC}, which now is interface-specific. Note that BW_{CAC} does not necessarily take up the whole reserved bandwidth BW. The reason for this is explained in detail in section 5.2.2.2 when describing the *II/E-CAC* scheme.



a) Node model for *ID-CAC* b) Possible flow distribution

Figure 5.4: CAC at ingress per outgoing interface (*II-CAC*)

Possible Realization

II-CAC could be realized by employing RSVP as the reservation protocol between end systems [BZB+97] and activating the RSVP feature only in the ingress routers. This way, each ingress router along the path between a sender and a receiver could evaluate the PATH messages and the corresponding RESV messages. As suggested in the IntServ framework, the reservations could be done on a per-outgoing-interface basis [BCS94]. However, there is a major constraint

with this realization. RSVP requires RESV messages to travel along the opposite routes of the corresponding PATH messages. In homogeneous IntServ environments, the routers take care of this. They store the number of the incoming interface for each PATH message and then send out the corresponding RESV request on this interface. If there are gaps with non-RSVP routers in between the RSVP-enabled ingress nodes, it has to be assured otherwise that RESV messages pass the same RSVP routers as their corresponding PATH messages. This is the case if in a network domain the ingress router is also the appropriate egress router for all IP addresses, which would enter the domain through this ingress node.

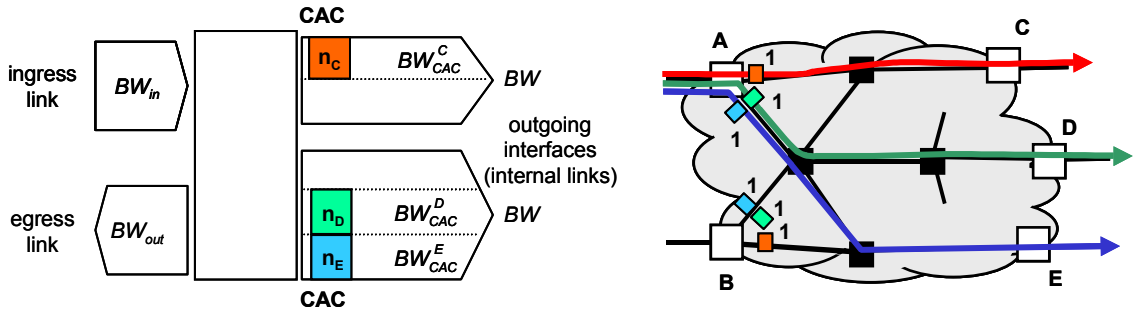
Consequences for Network Dimensioning

This CAC scheme reduces the potential spread of the traffic flows and, thus, provides a tighter worst-case flow pattern. In the example of Figure 5.4b only one connection can be set up between node A and node C. This is the limit, which the CAC instance enforces on the the corresponding outgoing interface. On the second interface, two simultaneous calls are permitted at maximum, leaving still some room for traffic uncertainties within the network. The two calls could either go to the same or to different egress nodes. Assuming an analog CAC scheme at node B, up to three concurrent connections might pass nodes C and E. For node D there is even the possibility that four connections are active at the same time (two from each of the ingress nodes).

5.2.1.3 CAC per Destination (*ID-CAC*)

Taking the preceding concept a step further, the CAC unit at the ingress router might not only consider the outgoing interface, on which the call enters the network, but rather the egress node of the call. A CA-controlled bandwidth component is associated with every egress node. Then, call admission is performed on basis of this destination-specific CA-controlled bandwidth (*ID-CAC*).

However, only locally available information is taken into account. In order for the ingress router to consider the egress location of a call, it needs to be able to map the call's destination address to a certain egress node. As the destination address might lie outside the scope of the domain, a methodolgy for this mapping has to be provided. The corresponding CAC model is depicted in Figure 5.5a.



a) Node model for *ID-CAC* b) Possible flow distribution

Figure 5.5: CAC at ingress per destination (*ID-CAC*)

Possible Realizations

One way to implement this form of CAC within an IP network is to employ MPLS and establish end-to-end label-switched paths (LSP) for every ingress-egress pair. Each LSP could then have a certain bandwidth assigned to it, which is controlled by the CAC instance at the ingress node. This way only a certain number of calls can be destined to a specific egress node.

Consequences for Network Dimensioning

From a dimensioning perspective this solution is very tight. Only a very specific number of simultaneously active calls can go from a certain ingress node to a specific egress node. There is no room for traffic flow variations. In the given example (Figure 5.5b) one call is allowed between each ingress-egress node pair giving a maximum of two active calls per egress node.

5.2.2 Call Admission Control at Ingress and Egress

So far, all CAC decisions are performed at one point in the network, only with the knowledge that is available at the ingress. However, egress links might not be dimensioned in a way that they can cope with the worst-case traffic. As a consequence, temporary overload situations at egress routers might lead to service degradation. To avoid traffic overload at the egress link, a CAC decision could also take into account the available bandwidth there. The three CAC schemes, which apply only control at the ingress, could be extended by CAC units at the egress, giving following alternatives:

- CAC per Ingress Link and Egress Link (*I/E-CAC*)
- CAC per Ingress Router Interface and Egress Link (*II/E-CAC*)
- CAC per Destination and Egress Link (*ID/E-CAC*)

Whenever a call setup is requested between a specific pair of ingress and egress nodes, the respective resources are checked at both locations. If one of the two bandwidth components is already fully utilized, the new call is blocked.

5.2.2.1 CAC per Ingress Link and Egress Link (*I/E-CAC*)

This CAC scheme corresponds to *I-CAC*, which is now extended by an additional admission check at the egress. The respective CA-controlled bandwidth components are the bandwidth of the ingress links and the bandwidth of the egress links (*I/E-CAC*).

Possible Realizations

The *I/E-CAC* scheme provides an abstraction for several IP-based CAC concepts, which have been proposed over the past years. Its relative simplicity makes it quite attractive for the Internet community.

One such concept has been developed in the European IST project AQUILA [EGK+03], where admission control is only one issue among many others. The access to a DiffServ network is regulated by admission control agents at the network edge in cooperation with agents at the egress. Before a call can be set up by an end system, it has to be signaled to

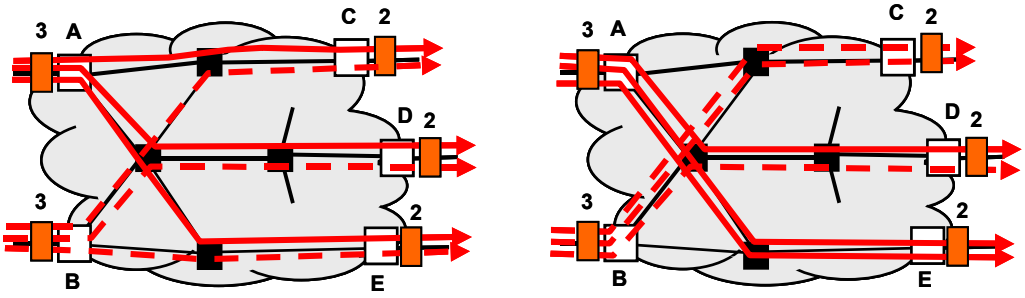


Figure 5.6: CAC per ingress link in coordination with egress link (*I/E-CAC*)

the control agent at the ingress, which then contacts the agent at the appropriate egress link. Only if both agents accept the call, it can be established.

Another concept, which also falls under the *I/E-CAC* scheme, is softswitch-based networking [Int03]. The functionality, which used to be integrated in a monolithic telecom switch, is now broken up and distributed over several systems. The interfaces between domains are provided through media gateways and the admission control functionality is taken over by a central unit, the media gateway controller. It is contacted every time a new call needs to be established. If for a call request the media gateways at the ingress and the egress of the domain still have enough free resources, the call is accepted. Otherwise, it is blocked. Thus, the CA-controlled bandwidth components are the capacities of the media gateways at the edges of the domain. They determine the number of simultaneously active calls, which can enter and leave the network at a certain location.

A third possibility of realizing *I/E-CAC* is an H.323 system with gatekeepers being configured with a certain limit of active connections originating in their region. However, this time (as opposed to the realization of *I-CAC*) the *I/E-CAC* scheme only works if the applications require bidirectional connection setup (such as VoIP). The originating H.323 terminal first makes a request to its own gatekeeper, which admits the call only if its CAC counter has not reached the limit, yet. Thereafter, the originating terminal signals the request to the destination terminal, which also has to setup a connection in order to establish the bidirectional communication channel. To do so, it has to contact its gatekeeper, which now can block the call if the reserved resources are already in use. Thus, the overall procedure represents a form of ingress-egress CAC where calls are only established if both gatekeeper accept the call. However, without any inter-domain signaling, this realization can only be applied in single domains.

Consequences for Network Dimensioning

Having admission control also at the egress nodes reduces the amount of required bandwidth at each egress link. Since now the number of outgoing calls can be controlled, it is not necessary to be able to cope with the worst-case flow patterns of the corresponding ingress-only CAC scheme. However, the two sample flow scenarios shown in Figure 5.6 indicate a still important issue for the network dimensioning procedure: although the maximum number of incoming and outgoing calls is fixed at each edge location, there are still different possibilities of flow distribution within the network. These have to be considered when dimensioning the

links if packet-level QoS degradation should be avoided. Nevertheless, due to the additional CA-controlled bandwidth component at the egress, the possible range of the flow distribution (i.e., the traffic uncertainty) is restricted. Therefore, it is expected that domain-internal links can be dimensioned more cost-efficiently.

5.2.2.2 CAC per Ingress Router Interface and Egress Link (*II/E-CAC*)

Analog to the preceding CAC scheme, *II-CAC* can also be extended by additionally carrying out admission control at the egress links of the domain. In this case, the CA-controlled bandwidth components are the configured bandwidth shares BW_{CAC} at the outgoing interfaces of ingress routers as well as the service-specific bandwidth BW_{out} of the egress links (*II/E-CAC*). For each call, a coordinated CAC decision is made.

The general principle of a *II/E-CAC* node is illustrated in Figure 5.7. This node implements ingress and egress functionalities, and also acts as a transfer node. All BW components in the figure indicate bandwidth shares, which are reserved exclusively for the considered traffic classes. These bandwidth shares take up either the whole link capacity or just a certain portion of it. For the presented dimensioning procedure later on, it is important to understand the necessity and significance of the various notions of bandwidth. Therefore, we discuss the abstracted node architecture in detail at this point. It is especially crucial to identify the bandwidth components that are affected by active connections. It should be noted that the following considerations apply analogously to all other CAC schemes. Only the BW_{CAC} components have to be changed according to the ingress schemes defined above.

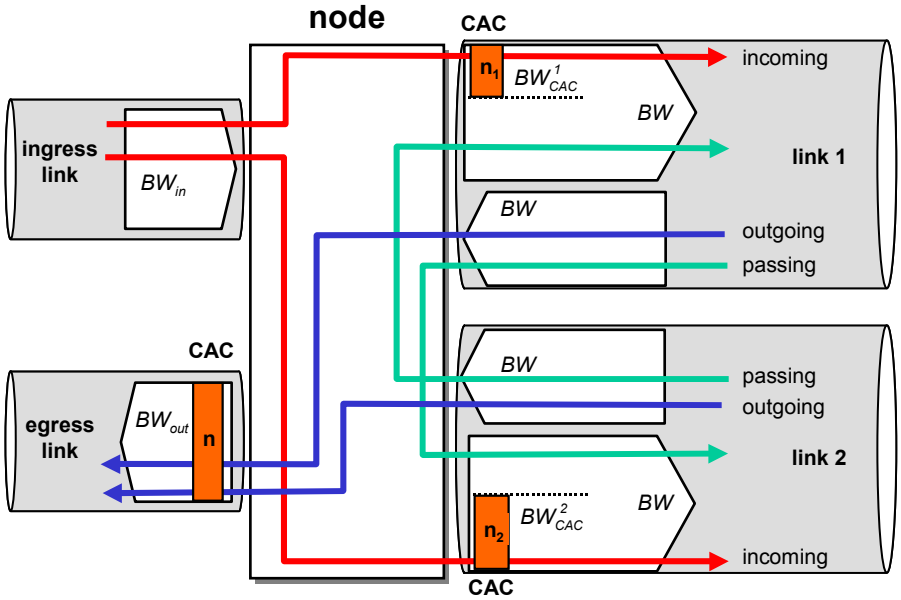


Figure 5.7: Node model for *II/E-CAC*

In general, a node can be traversed by a connection in three different ways:

- The connection enters the domain at this node (*incoming connection*).
- The connection leaves the domain at this node (*outgoing connection*).

- The connection passes this node within the domain (*passing connection*).

We do not consider cases where a connection passes the node from the "outside", i.e., where its packets first traverse the ingress link and, then, right away the egress link. The bandwidth components that are illustrated in the figure are only provided for traffic flows, which indeed go through the domain.

Incoming Connection If a connection enters the domain at a specific node, it first traverses the respective ingress link. On this link, it consumes a share of the overall reserved bandwidth BW_{in}. Since *II/E-CAC* is applied, the connection also occupies part of the CA-controlled bandwidth BW_{CAC} at the respective outgoing interface. The utilization of this bandwidth component was considered as basis for the CAC decision at the time when the connection was admitted into the network. On the way to its destination, the connection might pass several other nodes before it finally leaves the system at the appropriate egress node. The amount of bandwidth, which is utilized by the connection on each link, is equal to its bit rate.

Outgoing Connection An outgoing connection enters the node through an internal link where it takes up part of the traffic-class specific bandwidth BW. On the egress link, a portion of BW_{out} is consumed. As we consider an ingress-egress based CAC scheme, the bandwidth of the egress link is also subject to admission control. Thus, the connection could only have been established if at the time of call set up a bandwidth share equal to the required bit rate was still available.

Passing Connection In contrast to the two preceding connection types, a passing connection is not subject to admission control at the considered node at all. It reaches the node through an internal link and leaves it the same way (only on a different internal link). The CAC process for this connection did not taken into account any bandwidth components at this node, since it is neither the ingress nor the egress node of the connection. Therefore, the connection consumes shares of the reserved bandwidth BW at each link but is not part of any CA-controlled bandwidth component BW_{CAC}. CAC decisions were done based on bandwidth components at other nodes.

Possible Realizations

A QoS architecture with II/E-CAC could be realized through an overlay IntServ concept. RSVP-enabled routers are placed at the ingress and at the egress of the domain and are connected by IP-tunnels. In this case, a PATH message, which is sent out by the source, passes both routers. For the corresponding RESV message it has to be assured that the IP packet reaches the same RSVP-enabled routers. This can be done by setting up an explicit tunnel between the two routers. However, it is also possible without tunneling, if the reverse path runs over the same nodes as the forward path or if for all IP addresses the ingress router is also the appropriate egress router (see also realization of *II-CAC*).

Consequences for Network Dimensioning

The *II/E-CAC* scheme provides even tighter limits for the flow distribution within the network. While in the *I/E-CAC* case only one CA-controlled bandwidth existed for all incoming

connections, we have now again one per outgoing interface. Thus, the potential range of traffic distribution is further restricted. Figure 5.8a illustrates the *II/E-CAC* scheme.

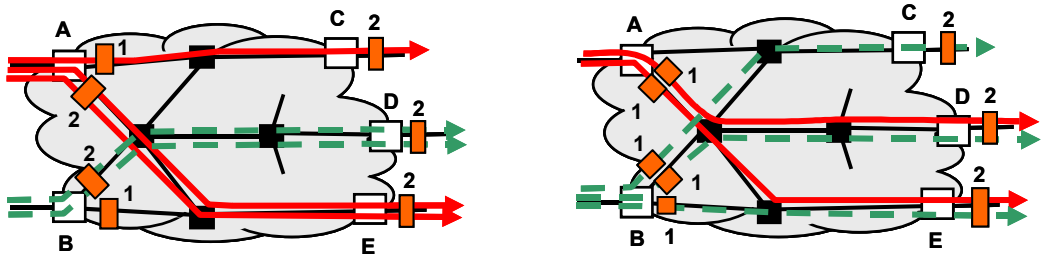


Figure 5.8: Ingress/egress CAC schemes per interface and per destination

5.2.2.3 CAC per Destination and Egress Link (*ID/E-CAC*)

The *ID/E-CAC* scheme is the consequent extension of *ID-CAC*. The CA-controlled bandwidth components are destination-specific bandwidth shares at the ingress routers and the bandwidth of the egress links. This scheme is a rather hypothetic case as it does not have much advantage over *ID-CAC*. However, its realization is more complex as the utilization has to be checked at two locations. The concept is depicted in Figure 5.8b.

Possible Realizations

Call admission control following *ID/E-CAC* would again best be implemented with MPLS and bandwidth-limited LSPs (same as in the case of *ID-CAC*). Only now, CAC decisions need to take into account information about utilization at the ingress as well as at the egress nodes.

Consequences for Network Dimensioning

ID/E-CAC provides a similarly tight bound on the potential traffic flow distribution within the network as the *ID-CAC* scheme. The difference between the two when it comes to network dimensioning is in the calculation of the CA-controlled bandwidth values. Having two CAC locations per flow requires a different approach than if CAC is only performed at one instance. However, this discussion is deferred to later sections.

5.2.3 Call Admission Control at all Nodes

The last CAC scheme, which is considered in this thesis, corresponds to the resource control scheme traditionally used in circuit-switched networks. A new call is only accepted if all links along the route from the ingress link to the egress link provide enough free bandwidth to accommodate the bit rate of the new connection. This scheme represents the most complex scenario in terms of CAC. For every arriving call all respective link loads have to be checked. Figure 5.9a illustrates the location of CA-controlled bandwidth components at each node.

The corresponding network scenario is depicted in Figure 5.9b. For further references this scheme is denoted as *All-CAC*.

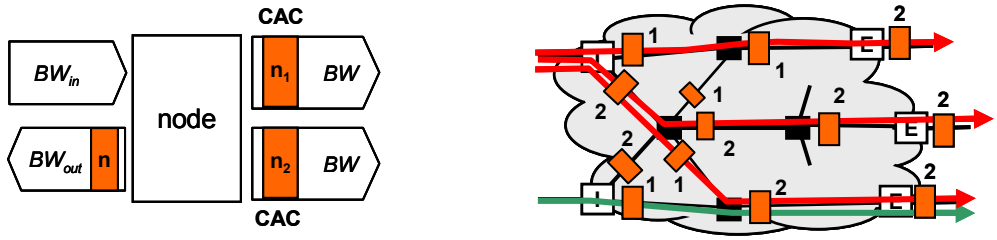


Figure 5.9: CAC at all nodes (*All-CAC*)

Possible Realizations

Several concepts exist, which apply this CAC scheme in IP networks. It can be implemented either in a distributed way as suggested by IntServ or in a central manner as proposed by various resource manager concepts [GM02, SP98]. IntServ requires a source to signal its reservation setup request (e.g., through RSVP), and every router along the path has the possibility to block it. Thus, a call is only accepted if every router along the way still has enough free bandwidth to carry the call. Resource management concepts, on the other hand, promote central servers, which know about the state of the network. Every time a new call should be established, these central servers are contacted and asked for permission. Although the practical realizations are very different, from a perspective of network dimensioning they have the same impact.

Consequences for Network Dimensioning

Once the CA-controlled bandwidth components are dimensioned, there is no room for variations of the traffic distribution. Therefore, no extra bandwidth has to be provided within the network in order to guarantee packet-level QoS. The dimensioning procedure corresponds to methods developed for circuit-switched networks such as PSTN or ATM.

5.2.4 Summary

The presented CAC abstractions cover the most important concepts, which have been proposed for IP networks. While some more variations of CAC schemes are conceivable (e.g., CAC at the egress instead of the ingress), the dimensioning approach would not be much different. The procedure for ingress-based CAC schemes, for example, could similarly be applied to egress-only CAC schemes. The fundamental ideas and calculations are the same.

Table 5.1 gives an overview of the CAC schemes and summarizes their main characteristics. The assessment of the complexity takes into account information availability, signaling, and configuration effort. All ingress-only CAC schemes are considered to be less complex than their ingress-egress counterparts since they do not require a coordinated CAC decision. The difference in complexity within the category of ingress-only schemes arises from the need for

	I-CAC	*II-CAC*	*ID-CAC*	*I/E-CAC*	*II/E-CAC*	*ID/E-CAC*	*All-CAC*
Location of CAC	only at ingress			coordinated at ingress and egress			every router
CA-controlled bandwidth at ingress	ingress link	outgoing interfaces	virtual, per destination	ingress link	outgoing interfaces	virtual, per destination	outgoing interfaces
Complexity	low	low/med	medium	med/high	high	high	very high

Table 5.1: Summary of considered CAC schemes

more sophisticated decision processes for *II-CAC* and *ID-CAC* in comparison to *I-CAC*. Analog considerations go for ingress-egress schemes. The *All-CAC* scheme is the most complex CAC architecture as it requires the coordination of the greatest number of CAC instances. If this coordination should be avoided, one central unit could take over the CAC decision for the whole network. However, then it has to be guaranteed that this instance is always informed about topology changes and configuration, as well as every active connection in the network.

5.3 Network Dimensioning Procedure

After having discussed the various CAC schemes and their abstractions, we are now able to present the optimization problem in detail. At first, the optimization model is considered in general, being valid for each type of CAC scheme. In the following, the procedure is applied to the various CAC schemes and the exact formulae are derived.

5.3.1 Optimization Problem

A network domain is again modeled as a directed graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with node set $\mathcal{V}$ and edge set $\mathcal{E}$. We refer to the nodes and edges by means of their indices. The nodes represent the routers of the network, while the edges represent the domain-internal links. We introduce mapping functions $s(l)$ and $t(l)$, which return the source node and the target node of a link l, respectively. Each of the $|\mathcal{V}|$ nodes implements ingress as well as egress functionalities. Thus, traffic flows might enter or leave the network at any node location. The bandwidth of the ingress and egress links at node i are denoted as $BW_{in}(i)$ and $BW_{out}(i)$, respectively. Each node can also serve as a transit node for flows, which do not originate or end at this node. The bandwidth allocated for the considered traffic class on link l is $BW(l)$ and the total amount of bandwidth in the domain is

$$BW_{total} = \sum_{i \in \mathcal{V}} BW_{in}(i) + \sum_{i \in \mathcal{V}} BW_{out}(i) + \sum_{l \in \mathcal{E}} BW(l) \tag{5.1}$$

With our assumption that the bit rates r are additive, we have $BW(l) = r \cdot N(l)$, with $N(l)$ being the number of simultaneous flows supported by link l (i.e., the number of *connection bit rate channels* on l). Bandwidths BW_{in} and BW_{out} can be specified accordingly.

The traffic demand, which is usually given in form of a traffic matrix, is defined as a set $\mathcal{F}$ of unidirectional demands. Each element of $\mathcal{F}$ specifies the traffic in Erlang a_f, $f \in \mathcal{F}$, which is being offered to the network between an ingress and an egress node. The nodes where demand f enters and leaves the domain are denoted as $o(f)$ (for *origin*) and $d(f)$ (for *destination*), respectively. In all cases, it is assumed that $o(f) \neq d(f)$. It is not required that a demand exists for all node pairs. If there is none specified, it is assumed that no connection is

node set	$\mathcal{V}$
link set	$\mathcal{E}$
demand set	$\mathcal{F}$
source node of link l	$s(l)$
target node of link l	$t(l)$
bandwidth of ingress link at node i	$BW_{in}(i)$
bandwidth of egress link at node i	$BW_{out}(i)$
bandwidth of internal link l	$BW(l)$
ingress node of demand f	$o(f)$
egress node of demand f	$d(f)$
set of links along the route of demand f	P_f
offered traffic of demand f	a_f
end-to-end call blocking probability of demand f	B_f
maximum number of active calls at respective links	$\hat{N}(l)$, $\hat{N}_{in}(i)$, $\hat{N}_{out}(i)$
vector of CA-controlled bandwidth components	$\mathbf{N}_{CAC}$
traffic flow distribution vector	$\mathbf{n}$

Table 5.2: Parameters and variables of network dimensioning for stream traffic

ever established between the two respective nodes. Based on a fixed routing pattern without load sharing, a set P_f can be determined for every demand f, containing all links along the route from ingress node $o(f)$ to egress node $d(f)$. Table 5.2 summarizes the parameters and variables of the network dimensioning problem.

Due to the application of CAC, the flows of demand f experience a certain end-to-end call blocking probability (EEB) B_f. This probability depends on the CAC scheme and the dimensioning of the CA-controlled bandwidth shares. In order to provide a certain connection-level QoS, this blocking probability should not exceed a given threshold B_{max}. In case that the domain is just one part of a sequence of domains, the overall QoS is affected by all of these networks. Then, it is best to break down the overall QoS requirement B_{e2e} and have every provider sign an agreement, which states that a certain domain-QoS is achieved. The individual domain blocking probability B_{max}, which every domain needs to comply with, can be approximated by $B_{max} = 1 - \sqrt[k]{1 - B_{e2e}}$, with k being the number of sequential domains.

Objective and Constraints

It is the objective of the dimensioning procedure to determine the bandwidth shares, which have to be allocated to the real-time service on every link in case one of the presented CAC schemes is implemented. The dimensioning procedure needs to consider the ingress and egress links as well as all network-internal links. It should be guaranteed that the desired QoS is

granted to a customer over the whole time span of a call, without service degradation periods.

The dimensioning procedure aims at the minimization of BW_{total} while still meeting the QoS requirements, i.e.,

$$z = \min \left(\sum_{i \in \mathcal{V}} BW_{in}(i) + \sum_{i \in \mathcal{V}} BW_{out}(i) + \sum_{l \in \mathcal{E}} BW(l) \right) \tag{5.2}$$

such that

$$B_f \leq B_{max} \qquad \forall\ f \in \mathcal{F} \tag{5.3}$$

$$BW_{in}(i) \geq r \cdot \hat{N}_{in}(i) \qquad \forall\ i \in \mathcal{V} \tag{5.4}$$

$$BW_{out}(i) \geq r \cdot \hat{N}_{out}(i) \qquad \forall\ i \in \mathcal{V} \tag{5.5}$$

$$BW(l) \geq r \cdot \hat{N}(l) \qquad \forall\ l \in \mathcal{E} \tag{5.6}$$

where $\hat{N}_{in}(i)$, $\hat{N}_{out}(j)$, and $\hat{N}(l)$ are the maximum possible numbers of simultaneously active flows on ingress link i, egress link j, and network-internal link l, respectively.

Constraints (5.3) state the connection-level QoS requirements. For each demand, the average blocking probability, which is experienced on the path through the domain, should not exceed a threshold B_{max}. In each CAC scheme, admission control is based on dedicated CA-controlled bandwidth shares. New calls are only blocked whenever these bandwidth shares are already fully utilized. As a consequence, the overall blocking probability is influenced solely by the CA-controlled bandwidth components and not by any other link bandwidth in the network. In order to stay below a certain threshold, the CA-controlled bandwidth components have to be dimensioned appropriately.

Expressions (5.4)-(5.6) assure that the desired packet-level QoS is guaranteed under any circumstances by dimensioning the links so they can carry the worst-case traffic. This requires the determination of the maximum possible number of active connections for each ingress, egress as well as internal link. As discussed in the preceding section, all possible traffic distribution patterns have to be taken into account. These depend on the employed CAC scheme and the dimensioning of the CA-controlled bandwidth components.

Outline of the Procedure

In order to describe the dimensioning procedure in detail, it is necessary to define a few more variables and reformulate the optimization problem. For readability purposes, we will mostly use the number of connection bit rate channels N when referring to a bandwidth value $BW = N \cdot r$.

First, it is important to notice that the total network bandwidth can be uniquely derived from a specific setting of the CA-controlled bandwidth components. Although these bandwidth shares do not appear explicitly in the objective function and in the constraints given above, they directly determine all other bandwidth values in the network. Therefore, the CA-controlled bandwidths are used as optimization variables. We specify a vector $\mathbf{N}_{CAC}$, whose elements are the number of connection bit rate channels at each CAC location (as discussed in section 5.2). The exact interpretation of the vector depends on the considered

CAC scheme. Its dimension is equal to the number of CA-controlled bandwidth components in the network.

Furthermore, we define a vector $\mathbf{n} \in \mathbb{N}_0^{|\mathcal{F}|}$, which reflects a possible traffic situation at a certain time. It contains one entry $n(f)$ for each demand $f \in \mathcal{F}$, which is a positive integer, including 0, and which represents the number of active connections along the path of demand f. While the average offered load of demand f is a_f (measured in Erlang), the actual number of connections varies over time, following a Markov process. The maximum of each $n(f)$ depends on the dimensioning of the CA-controlled bandwidth components, which are located along the path of demand f. Furthermore, the individual $n(f)$ values are interrelated since many demands share common CA-controlled bandwidth components, which limit the total number of active connections. As a consequence, a specific setting of $\mathbf{N}_{CAC}$ restricts the space of vector $\mathbf{n}$. For further references to this type of restriction, we introduce notation $\mathbf{n} \dashv \mathbf{N}_{CAC}$ (speak "$\mathbf{n}$ *complies with* $\mathbf{N}_{CAC}$"), which means that the specific traffic situation represented by $\mathbf{n}$ complies with the CA-controlled bandwidth setting $\mathbf{N}_{CAC}$. It should be emphasized that the restriction does not apply on an element-by-element basis. It depends on the network topology, the routing configuration, and the type of CAC scheme. As we have seen in Section 5.2, the various CAC schemes provide restrictions of different tightness.

With the additional variables and the new notation, the optimization problem can be reformulated as follows:

$$\tilde{z} = \min_{\mathbf{N}_{CAC}} \left(\sum_{i \in \mathcal{V}} \hat{N}_{in}(i) + \sum_{i \in \mathcal{V}} \hat{N}_{out}(i) + \sum_{l \in \mathcal{E}} \hat{N}(l) \right) \tag{5.7}$$

with

$$\hat{N}_{in}(i) = \max_{\mathbf{n}} \left(N_{in}(i) : \mathbf{n} \dashv \mathbf{N}_{CAC} \right) \qquad \forall\, i \in \mathcal{V} \tag{5.8}$$

$$\hat{N}_{out}(i) = \max_{\mathbf{n}} \left(N_{out}(i) : \mathbf{n} \dashv \mathbf{N}_{CAC} \right) \qquad \forall\, i \in \mathcal{V} \tag{5.9}$$

$$\hat{N}(l) = \max_{\mathbf{n}} \left(N(l) : \mathbf{n} \dashv \mathbf{N}_{CAC} \right) \qquad \forall\, l \in \mathcal{E} \tag{5.10}$$

such that

$$B_f(\mathbf{N}_{CAC}) \leq B_{max} \qquad \forall\, f \in \mathcal{F} \tag{5.11}$$

Thus, the objective function is the minimization of the required connection bit rate channels over variables $\mathbf{N}_{CAC}$, where the individual terms of the sum have to be again determined by means of optimization. To find the worst-case traffic on each link the maximization has to be carried out over all traffic load patterns $\mathbf{n}$, which comply with the optimization variables $\mathbf{N}_{CAC}$ (5.8)-(5.10).

As it is not possible to solve the optimization problem in the given way, we propose a two-step procedure. The overall problem is divided into two sub-tasks, which are then solved independently:

1. **Dimensioning of CA-controlled bandwidths**
   At first, the CA-controlled bandwidth shares are dimensioned without taking into consideration any other bandwidth values. This corresponds only to part of the optimization problem given above. The reduced optimization problem comprises an objective

function $\tilde{z}^* = \min\left(\sum_k N_{CAC}\left(k\right)\right)$ and constraints (5.11). As a result, the amount of CA-controlled bandwidth in the network is minimized, while the connection-level QoS requirements are met.

2. **Robust dimensioning for load variations**
   In a second step, the remaining bandwidth values are computed as specified in Equations (5.8)-(5.10). The focus of this step is the robust dimensioning of all links so that packet-level QoS requirements are not violated under any flow-related circumstances.

The rationale behind our procedure is the following: In the first step we minimize the traffic inflow, i.e., the amount of traffic that is accepted to traverse our network domain. The less traffic that is allowed to enter the network, the less bandwidth will be needed within the network to provide the desired QoS. Thus, the first step is focusing on saving bandwidth. In the second step, we have to make the domain robust against all possible flow patterns in order to avoid service degradation of active calls. Here, the issue is not about saving bandwidth but about guaranteeing QoS.

5.3.2 Dimensioning CA-controlled Bandwidths

When dimensioning the CA-controlled bandwidth shares, we have to differentiate between CAC schemes that base their decisions only on information available at the ingress (*I-CAC*, *II-CAC*, *ID-CAC*) and those, which carry out coordinated admission control (*I/E-CAC*, *II/E-CAC*, *ID/E-CAC*, *All-CAC*). For the first group Erlang's first formula ("Erlang B") can be applied directly. For the cases, where CAC is performed in a coordinated fashion, the dimensioning procedure of CA-controlled bandwidths becomes trickier. However, it turns out that for all coordinated CAC schemes this procedure corresponds to the dimensioning problem for circuit-switched networks such as PSTN or ATM. In the following we present the solution procedure for all CAC schemes in detail.

5.3.2.1 CAC at Ingress

The appropriate CA-controlled bandwidth shares are computed by numerically inverting Erlang's blocking formula $B_{max} = E(A, N)$. Parameter A is the amount of traffic (in Erlang) offered to the respective bandwidth, which is either an ingress link, an outgoing link interface, or a per-destination virtual hop, and N denotes the number of connection bit rate channels required to handle the offered load.

I-CAC

For *I-CAC* schemes, the CA-controlled bandwidth components are the ingress links of the network, i.e., $\mathbf{N}_{CAC} = \mathbf{N}_{in}$ where $\mathbf{N}_{in}$ is the vector containing all $N_{in}(i)$ with $i \in \mathcal{V}$. The offered load $A(i)$ per CA-controlled component $N_{in}(i)$ is the total demand, which enters the network at node i. Thus, the bandwidth shares of the ingress links are calculated by solving following equations:

$$A(i) = \sum_{f \in \mathcal{F}: o(f) = i} a_f \qquad \forall\, i \in \mathcal{V} \tag{5.12}$$

$$\hat{N}_{in}(i) = \arg\left\{ B_{max} = E\left(A(i), \hat{N}_{in}(i)\right)\right\} \qquad \forall\, i \in \mathcal{V} \tag{5.13}$$

$$BW_{in}(i) = \hat{N}_{in}(i) \cdot r \qquad \forall\ i \in \mathcal{V} \tag{5.14}$$

II-CAC

II-CAC schemes apply call admission control at the outgoing interfaces of all ingress nodes. Therefore, we have to first determine the amount of offered load $A(l)$ on each ingress node's outgoing interface. Note that for the general assumption that every node can be an ingress link, every link l in the network is also an outgoing interface of some ingress node i, namely of node $i = s(l)$. Based on the respective load values, the CA-controlled bandwidths can be computed. Following equations are relevant:

$$A(l) = \sum_{f \in \mathcal{F}: o(f) = s(l) \wedge P_f \ni l} a_f \qquad \forall\ \ l \in \mathcal{E} \tag{5.15}$$

$$\hat{N}_{CAC}(l) = \arg \left\{ B_{max} = E\left(A(l), \hat{N}_{CAC}(l)\right) \right\} \qquad \forall\ l \in \mathcal{E} \tag{5.16}$$

$$BW_{CAC}(l) = \hat{N}_{CAC}(l) \cdot r \qquad \forall\ l \in \mathcal{E} \tag{5.17}$$

ID-CAC

The *ID-CAC* scheme does not consider any link specific bandwidths for call admission control. CAC is applied to ingress-egress relationships, similar to virtual channels. Therefore, we can define one CA-controlled bandwidth related to every demand f. The offered load is a_f.

$$\hat{N}_{CAC}(f) = \arg \left\{ B_{max} = E\left(a_f, \hat{N}_{CAC}(f)\right) \right\} \qquad \forall\ f \in \mathcal{F} \tag{5.18}$$

$$BW_{CAC}(f) = \hat{N}_{CAC}(f) \cdot r \qquad \forall\ f \in \mathcal{F} \tag{5.19}$$

5.3.2.2 CAC at Ingress in coordination with Egress

A network, in which CAC is applied at the ingress nodes in coordination with egress links, can be transformed into a virtual network, which contains only CA-controlled bandwidth components. It is important to notice that every traffic flow traverses exactly two bandwidth components, which are subject to CAC. These are either the ingress links, the outgoing interfaces, or the virtual ingress/egress hops together with the egress links. Other than that, a connection will not experience any blocking. Therefore, the original network can be represented by a star-structured network with one non-blocking center node and a number of unidirectional edges. Figure 5.10 illustrates this transformation, by showing the center node as a non-blocking, fully-meshed network cloud. The incoming edges represent either the ingress links (*I/E-CAC*), the outgoing interfaces of all ingress routers (*II/E-CAC*), or all virtual per-destination hops (*ID/E-CAC*). The outgoing links of the center node correspond to the egress links of the domain. Thus, the total number of edges in the transformed network equals the number of elements of vector $\mathbf{N}_{CAC}$. Each link m of the transformed network has a capacity C_{in}^m or C_{out}^m assigned to it, which represents a CA-controlled bandwidth value of the original network. Since a connection traverses two links, the overall blocking probability

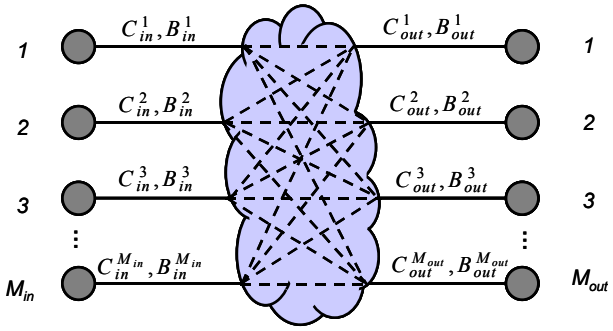


Figure 5.10: Network transformation for ingress-egress CAC schemes

of each connection is a function of two link-specific blocking probabilities denoted as B_{in}^m or B_{out}^m.

The dimensioning task corresponds to the nonlinear capacity assignment problem where the sum of the capacities has to be minimized while the end-to-end delay for each traffic flow is kept below a certain limit. Thus, we can dimension the network applying methods known from PSTN or ATM network planning.

In our approach the dimensioning problem is solved by taking the blocking probabilities as optimization variables and applying the reduce-load approximation. For a certain setting of blocking probabilities, the Erlang fixed-point equations can be set up and the amount of offered traffic can be derived for every link [Gir90]. From this, the bandwidth components can be computed. The overall optimization is performed by means of the conjugate-gradient method with penalty function [GMW93, NW99]. The interpretation of the results depends on the CAC scheme. At the end, we have following bandwidth values:

- ***I/E-CAC***: $BW_{in}(i) = \hat{N}_{in}(i) \cdot r,\ BW_{out}(i) = \hat{N}_{out}(i) \cdot r\ \forall\ \ i \in \mathcal{V}$
- ***II/E-CAC***: $BW_{CAC}(l) = \hat{N}_{CAC}(l) \cdot r\ \forall\ l \in \mathcal{E}$ and $BW_{out}(i) = \hat{N}_{out}(i) \cdot r\ \forall\ i \in \mathcal{V}$
- ***ID/E-CAC***: $BW_{CAC}(f) = \hat{N}_{CAC}(f) \cdot r\ \forall\ f \in \mathcal{F}$ and $BW_{out}(i) = \hat{N}_{out}(i) \cdot r\ \forall\ i \in \mathcal{V}$

5.3.2.3 CAC at all Nodes

This scenario is very similar to the preceding one. However, we do not need to transform the network in order to apply the optimization procedure. The scenario with CAC at all nodes is the same as the general network dimensioning problem for PSTN or ATM networks. The resulting capacity values correspond to the CA-controlled bandwidth shares on each network link. These are $BW_{in}(i) = \hat{N}_{in}(i) \cdot r,\ BW_{out}(i) = \hat{N}_{out}(i) \cdot r$ for $i \in \mathcal{V}$ and $BW_{CAC}(l) = \hat{N}_{CAC}(l) \cdot r$ for $l \in \mathcal{E}$.

5.3.3 Robust Dimensioning for Load Variations

After fixing the CA-controlled bandwidth shares, the remaining bandwidth values have to be determined. The aim is to assure packet-level QoS for all possible traffic distribution patterns by setting the bandwidth of every link large enough to carry the worst-case traffic. This step

requires the solution of the relevant maximization problems defined in equations (5.8)-(5.10). Ignoring the requirement $\mathbf{n} \dashv \mathbf{N}_{CAC}$ for the moment, we get the following equations

$$\hat{N}_{in}(i) = \max_{\mathbf{n}} \left(\sum_{f \in \mathcal{F}: o(f)=i} n(f) \right) \qquad \forall\, i \in \mathcal{V} \tag{5.20}$$

$$\hat{N}_{out}(i) = \max_{\mathbf{n}} \left(\sum_{f \in \mathcal{F}: d(f)=i} n(f) \right) \qquad \forall\, i \in \mathcal{V} \tag{5.21}$$

$$\hat{N}(l) = \max_{\mathbf{n}} \left(\sum_{f \in \mathcal{F}: P_f \ni l} n(f) \right) \qquad \forall\, l \in \mathcal{E} \tag{5.22}$$

These maximization problems are the same for every CAC scheme. However, some of the bandwidth variables are already fixed after the first step of the dimensioning procedure. Therefore, it is only necessary to solve one or two of the given problems. The main difference between the various CAC schemes lies in the impact of $\mathbf{n} \dashv \mathbf{N}_{CAC}$, which for each scheme is formulated as a set of linear constraints.

I-CAC

In case of *I-CAC* the first step of our dimensioning procedure returns the bandwidth shares of the ingress links. This means that all $\hat{N}_{in}(i)$ are fixed at this point. Therefore, we have to solve only the two optimization problems (5.21) and (5.22) with $\mathbf{n} \dashv \mathbf{N}_{CAC}$ being expressed as

$$\sum_{f \in \mathcal{F}: o(f)=i} n(f) \leq \hat{N}_{in}(i) \qquad \forall\, i \in \mathcal{V} \tag{5.23}$$

With (5.21) and (5.23) we take into account that all ingress traffic could be destined to one egress node. Objective function (5.22) together with (5.23) finds the number of flows, which could possibly traverse a certain internal link. Only if the link is capable of carrying this maximum number of flows, traffic overload and service degradation can be avoided at all times.

In case of ingress-only CAC schemes, the maximization problems can also be solved by inspection. For each bandwidth component, which has to be dimensioned, we have to look at the traffic demands that traverse it. The worst case situation is certainly when the corresponding connections take up the whole bandwidth, which they could possibly seize at their respective point of admission control. At this worst-case moment, no other connections going into other areas of the network are present at the respective CA-controlled bandwidth locations.

In an *I-CAC* network, the worst-case traffic at the egress link of node i is the sum of the bandwidth shares of all ingress links, for which a demand exists towards node i, i.e.,

$$\hat{N}_{out}(i) = \sum_{f \in \mathcal{F}: d(f)=i} \hat{N}_{in}\left(o(f)\right) \qquad \forall\, i \in \mathcal{V}$$

Similarly, the bandwidth of an internal link l is the sum of the ingress bandwidths of all nodes, which are the entry points of the demands that go over link l.

II-CAC

The *II-CAC* scenario is very similar to the preceding one. Only now, the first step of the dimensioning procedure does not return the bandwidth of the ingress links, but the CA-controlled bandwidth shares $\hat{N}_{CAC}(l)$ on the outgoing interfaces of each ingress router. One such bandwidth share can be associated with every link l. Since ingress routers can also be transit nodes, they might be traversed by connections, which enter and leave the domain at some other points. To accommodate these flows extra bandwidth has to be provided.

Thus, all three maximization problems (5.20)-(5.22) are relevant. Each one is subject to

$$\sum_{f \in \mathcal{F}: o(f)=s(l) \wedge P_f \ni l} n(f) \leq \hat{N}_{CAC}(l) \qquad \forall\ l \in \mathcal{V} \tag{5.24}$$

Solving optimization problems (5.20)+(5.24), (5.21)+(5.24), and (5.22)+(5.24) gives the required bandwidth values.

Again, the solutions can be found explicitly by inspection. For each ingress link at node i, $\hat{N}_{in}(i)$ is the sum of the CA-controlled bandwidths at the outgoing interfaces of node i. For the egress links and the internal links we have to trace back the traversing demands to the respective CA-controlled components and sum up the respective bandwidth values $\hat{N}_{CAC}(l)$.

ID-CAC

As discussed in Section 5.2.1.3, the *ID-CAC* architecture does not leave any room for traffic variations. Each traffic stream f is regarded as a virtual hop through the network, and a CA-controlled bandwidth $\hat{N}_{CAC}(f)$ is associated with it. Thus, $\mathbf{n} \dashv \mathbf{N}_{CAC}$ is specified through constraints

$$n(f) \leq \hat{N}_{CAC}(f) \qquad \forall\ f \in \mathcal{F} \tag{5.25}$$

For each link in the network, the required bandwidth is simply the sum of the CA-controlled bandwidth "pipes", which traverse the link. The final bandwidth components are:

$$\hat{N}_{in}(i) = \sum_{f \in \mathcal{F}: o(f)=i} \hat{N}_{CAC}(f) \qquad \forall\ i \in \mathcal{V}$$

$$\hat{N}_{out}(i) = \sum_{f \in \mathcal{F}: d(f)=i} \hat{N}_{CAC}(f) \qquad \forall\ i \in \mathcal{V}$$

$$\hat{N}(l) = \sum_{f \in \mathcal{F}: P_f \ni l} \hat{N}_{CAC}(f) \qquad \forall\ l \in \mathcal{E}$$

I/E-CAC

In *I/E-CAC* scenarios the bandwidth shares of the ingress links $\hat{N}_{in}(i)$ as well as the egress links $\hat{N}_{out}(i)$ are determined during the optimization of the first dimensioning step. The only links that are left for dimensioning are the network-internal links.

The fact that *I/E-CAC* limits the number of active flows at the ingress as well as at the egress links leads to two types of constraints. The ingress constraints are the same as in (5.23). These are now complemented by the set of egress constraints (5.26). To come up with the remaining bandwidth values $\hat{N}(l)$ we have to solve (5.22) subject to (5.23) and (5.26).

$$\sum_{f \in \mathcal{F}: d(f)=i} n(f) \leq \hat{N}_{out}(i) \qquad \forall\, i \in \mathcal{V} \tag{5.26}$$

II/E-CAC

The dimensioning of the CA-controlled bandwidth components for *II/E-CAC* returns the values $\hat{N}_{CAC}(l)$ for every outgoing interface at the ingress routers (each interface being associated with a link l) as well as again the bandwidth of each egress link $\hat{N}_{out}(i)$. Therefore, the relevant set of constraints are (5.24) and (5.26) and the problems to solve are (5.20) in order to dimension the ingress links $\hat{N}_{in}(i)$ and (5.22) for the internal links $\hat{N}(l)$.

ID/E-CAC

Following the reasoning of the *II/E-CAC* scheme, the relevant optimization problems for *ID/E-CAC* are again (5.20) and (5.22), this time subject to the constraints (5.25) and (5.26).

All-CAC

If an *All-CAC* scheme is implemented, the first step of the dimensioning procedure gives already all bandwidth values. Since every link is subject to admission control, there is no possibility of uncontrolled traffic variation anywhere in the network.

5.3.4 Extension for Multiple Egress Links

For the presented dimensioning strategy it has been assumed that there is only one egress link per egress router. Thus, only one CA-controlled bandwidth component had to be considered at each egress node. However, if there are several egress links per egress router, the traffic demands and the corresponding flows need to be differentiated accordingly. Similar to ingress alternatives, it is then conceivable to perform call admission control based on either the aggregate, the further direction of the flows (i.e., the outgoing egress link), or even their destination. For the dimensioning process this would mean that more CA-controlled bandwidth components have to be taken into account, reducing the trunking gain at the egress. Nevertheless, the network could again be transformed into a two-stage, non-blocking graph, whose link capacities are again optimized using the same algorithms. For the second dimensioning step, the constraints would have to be modified, analog to the presented ingress options. Allowing all CAC options at the egress, we would end up with new CAC schemes, such as *I/EI-CAC*, *I/ED-CAC*, and so forth.

5.4 Numerical Results and Discussion

The dimensioning procedure was applied to the sample network topologies listed in Appendix A. The traffic between the nodes was generated randomly. For all investigations a connection bit rate of 64 kbps is assumed and in all cases the end-to-end blocking probability in the domain is required to be below 1%.

Table 5.3 summarizes the bandwidth needs for the various CAC schemes. For each network the bandwidth total (in Mbps) is given as well as the individual terms for ingress, egress, and internal links. In the following paragraphs the key observations are discussed and highlighted.

		N11	**N20**	**N40**	**N50**	**N100**
I-CAC	total	**29080**	**52670**	**42586**	**154860**	**341934**
	ingress	1385	1285	1073	1564	1718
	egress	13848	24418	17578	76648	170108
	internal	13848	26968	23953	76648	170108
II-CAC	total	**9223**	**17407**	**24239**	**129250**	**265009**
	ingress	1442	1369	1109	1615	1831
	egress	3891	7596	9849	63818	131589
	internal	3891	8443	13281	63818	131589
ID-CAC	total	**5454**	**6087**	**6442**	**12967**	**27479**
	ingress	1510	1546	1370	2579	4535
	egress	1510	1546	1370	2579	4535
	internal	2435	2996	3703	7809	18409
I/E-CAC	total	**13065**	**12710**	**10248**	**17277**	**23510**
	ingress	1395	1297	1089	1586	1748
	egress	1395	1298	1089	1586	1748
	internal	10274	10115	8071	14105	20013
II/E-CAC	total	**6776**	**7617**	**7854**	**16133**	**18644**
	ingress	1454	1384	1124	1649	1884
	egress	1402	1306	1094	1581	1740
	internal	3921	4927	5636	12903	15020
ID/E-CAC	total	**5395**	**5913**	**6247**	**11155**	**20850**
	ingress	1532	1567	1390	2543	4307
	egress	1406	1312	1107	1623	1819
	internal	2457	3043	3751	6988	14723
All-CAC	total	**5158**	**5234**	**5305**	**8075**	**10664**
	ingress	1408	1312	1109	1611	1787
	egress	1408	1312	1109	1611	1787
	internal	2343	2610	3087	4853	7091

Table 5.3: Bandwidth requirements (in Mbps) for stream traffic

Figure 5.11 illustrates the results of the dimensioning process for each of the CAC schemes and each of the network scenarios. The bandwidth totals are normalized with respect to *All-CAC*. *All-CAC* requires the least amount of bandwidth as it carries out call admission control at each link and, therefore, achieves the maximum trunking gain. One has to be aware, though, that *All-CAC* is also the most complex scheme. Every link needs to be configured with the appropriate CA-controlled bandwidth share, and for admission decisions information about all links has to be available. On the other hand, the *I-CAC* and *II-CAC* schemes are the easiest ones to implement. However, they lead to very high bandwidth values. The numbers in the graph of Figure 5.11 mark the bandwidth requirements for the respective *I-CAC* scheme. For network *N100*, I-CAC requires for example over 32 times as much bandwidth as *All-CAC*. In all cases, the resulting overdimensioning is mainly due to the substantial bandwidth requirements of egress and internal links, necessary to assure robustness.

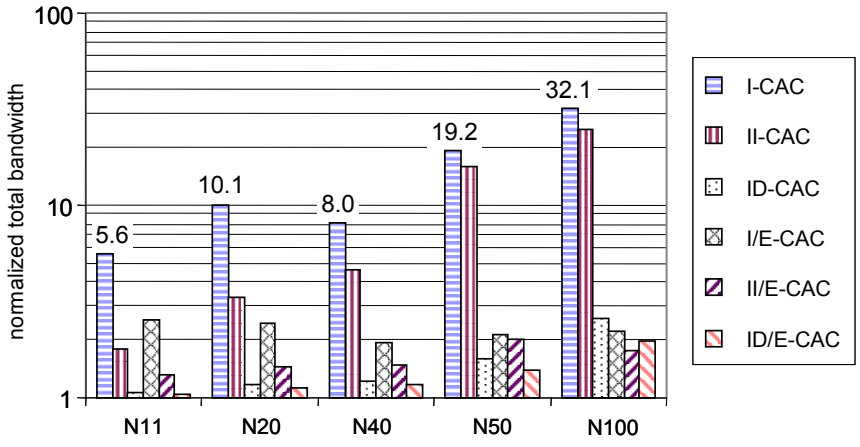


Figure 5.11: Normalized bandwidth total for different CAC schemes and various network scenarios

In the graph of Figure 5.12 the individual bandwidth portions are depicted for network scenario *N40*. It becomes clear that applying simple admission control can only be compensated by excessive overdimensioning. For *I-CAC* and *II-CAC* the required bandwidth values at internal links as well as at egress links are significantly higher than for any other CAC scheme. By introducing signaling between ingress and egress nodes, the bandwidth costs can be considerably reduced.

If one would like to achieve capacity-efficient CAC with signaling only at the ingress nodes, *ID-CAC* would be the scheme of choice. The bandwidth costs for this scheme come close to the costs of *All-CAC*. As a drawback, *ID-CAC* is less robust towards deviations of the traffic matrix than *I-CAC* or *II-CAC*. As the bandwidth pipes provide a very tight bound, small variations of the traffic load already lead to increased blocking probabilities for some demands. *I-CAC* and *II-CAC* can cope with variations of the demand matrix to a certain extent as long as the offered traffic does not increase in total.

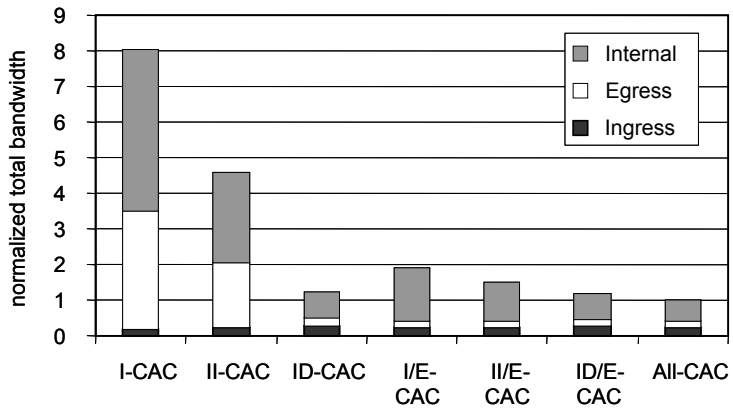


Figure 5.12: Normalized bandwidth components for network scenario *N40*

5.5 Summary

In this chapter, a dimensioning methodology for real-time services with hard QoS requirements was presented. We defined generic CAC schemes, which can be matched to most of the currently discussed technical CAC approaches. For each of these schemes a network dimensioning algorithm is presented and its applicability is demonstrated.

The results of the dimensioning procedure for the sample network scenarios clearly demonstrate that there exists a trade-off between the complexity of the call admission control scheme on the one side and overdimensioning on the other. The required bandwidth totals for the two simplest schemes are significantly higher. Nevertheless, for some network service providers, the simplicity of a CAC scheme might be worth more than the bandwidth savings.

Chapter 6

Conclusion and Outlook

The Internet Protocol family has emerged as the enabling technology for the convergence of information technology, telecommunications, and media distribution. For the future, *all-IP networks* are considered the ultimate solution for truly integrated-services networking. However, the past few years have shown that the realization of all-IP networking is more difficult than expected. Often, it is not the technical problems, but rather the economic considerations, which defer its evolution. The downturn of the economy has forced network service providers to reduce expenditures. In order to succeed in the coming years, tight profit margins will have to be compensated by efficient design and operation of network infrastructure.

The Need for Traffic Engineering and Network Planning

By analyzing the characteristics and implications of services and applications, which are expected to significantly influence the development and use of future IP technology, it becomes clear that the need for traffic engineering and network planning will be intensified.

The diverse and often strict demands of emerging applications regarding quality of service will require differentiated treatment of services within the network. It will be inevitable that QoS-enabling technologies are implemented, and that each service class will be provided with the appropriate resources needed to guarantee the desired QoS. This requires thorough planning of link capacities and bandwidth allocation schemes, taking into account the specifics of new services and technologies.

Furthermore, due to the uncertainty of new services and applications concerning traffic volume and characteristics, providers will not be able to accurately predict the load situation in the network before it is actually in operation. Therefore, service providers will have to stay flexible when it comes to service provisioning and bandwidth allocation. They need to be able to react to traffic variations and skewed load distributions in order to avoid localized congestion and overload situations. Traffic engineering comprises means and methodologies to alleviate this type of problems in IP networks.

This dissertation proposes algorithms and methodologies, which let providers optimize their network infrastructure in order to reduce expenditures, while still being able to offer a desired quality of service. We specifically focus on routing optimization and capacity assignment for multi-service IP networks as these are – and will be – two key issues of traffic engineering and network planning. For the capacity assignment process we distinguish be-

tween two basic types of traffic, elastic and stream traffic, which are associated with data applications and real-time services, respectively. Since their characteristics and QoS requirements are fundamentally different, it is necessary to come up with two distinct approaches for network dimensioning.

Routing Optimization

Routing optimization is mainly considered in the context of traffic engineering. It provides a means to balance the traffic load in the network in order to improve quality of service. Two fundamentally different strategies of routing optimization are investigated in this dissertation. In the first case, the optimization is based on conventional routing protocols such as OSPF and EIGRP, which are standard in every IP network. In the second case, MPLS is utilized as a complementary technology to improve an existing routing configuration. Since MPLS provides greater flexibility, it could be used to further enhance a solution, which has been optimized with OSPF or EIGRP. The main objective of all routing optimization methods is the minimization of the maximum link utilization.

Conventional routing protocols such as OSPF or EIGRP implement shortest-path routing. The calculation of the distances between all nodes is based on link metrics, which are set statically by the network administrator. By adjusting the values of these metrics (i.e., the *link weights*), the route selection process can be influenced and, thus, the path pattern optimized. A significant novelty of our approach is the consideration of additive as well as concave link metrics as supported by EIGRP. The combination of the two metric types allows a more flexible route selection and, therefore, provides a greater optimization potential. In any case, routing optimization with two metrics is always at least as good as routing optimization with one metric. The advantage of a dual-metric protocol over its single-metric counterpart depends on the network topology and the load situation.

The link weight setting problem is formulated as a mixed-integer linear program, which for small networks can be solved with standard tools. For larger networks, a hybrid genetic algorithm was developed. This algorithm does not only support the global optimization of routing, but also a notion of routing adaptation. Instead of trying to find the global optimum, one might be interested in improving a routing configuration while limiting the amount of rerouted traffic or the number of modified link weights. This could be important in networks where routing optimization is carried out frequently and rerouted flows experience service degradation due to transient states. Then, the rerouting impact should be kept as low as possible.

On the basis of several scenarios, the potential of routing optimization with OSPF (one metric) and EIGRP (two metrics) is demonstrated. In all cases, optimizing the link weights reduces the maximum link utilization substantially as compared to standard hop-based shortest path routing (i.e., all link weights are set to 1). For some scenarios, the final results are even very close to the theoretical lower bound, which is computed by solving a modified version of the linear multicommodity flow problem. It can be shown that the use of two metric types is indeed superior to one metric. In some scenarios routing optimization with EIGRP is considerably better than with OSPF, achieving additional gains of over 10%.

If further improvement is desired, MPLS has to be employed. With this technology it is possible to implement any path pattern since every end-to-end flow can be routed independently. Unlike most other approaches in the area of MPLS routing optimization, which

establish label-switched paths between all nodes of a network, we pursue a hybrid approach. The majority of traffic is still routed along shortest paths in native IP fashion and only a small number of flows is selected and forwarded along label-switched paths. This way, the administrational effort is lower since the number of paths, which need to be established explicitly, is small. Furthermore, as only the routers along the label-switched paths would have to support MPLS, the expenses for new equipment can be reduced. Essentially, our approach aims at optimized routing adaptation rather than global optimization.

For the MPLS routing optimization problem two mixed-integer linear programs are presented. Starting with a given configuration of shortest-path routing, an optimal set of flows is identified for rerouting and the corresponding paths are computed. One objective is again the minimization of the maximum link utilization under the constraint that only a certain number of flows are rerouted. However, other objectives can also be addressed. It is possible to minimize the number of rerouted flows or the number of affected routers, while keeping the utilization below a certain threshold.

By applying the MPLS routing adaptation methods to the sample network scenarios, it can be shown that considerable QoS improvement is possible if only a small number of label-switched paths are set up. In some networks establishing label-switched paths for as little as 5-10% of the flows is sufficient to reach the global optimum. In the same way, only a subset of all routers needs to be turned into MPLS routers in order to achieve respectable QoS enhancements.

Overall, it can be concluded that routing optimization provides a promising tool for network service providers to resolve localized overload situations and, thus, increase quality of service. As can be demonstrated in our scenarios, it is not always necessary to invest in new technologies. Conventional routing protocols already provide great possibilities for traffic engineering. If desired, routing can be improved with MPLS by setting up only few label-switched paths.

Dimensioning for Elastic Traffic

Elastic traffic is associated with data applications, which use TCP for data transfer. The characteristic property of an elastic traffic flow is its rate adaptability, which is caused by the feedback mechanism of TCP. The transmission rate of a sender is adjusted to the available bandwidth along the route towards the receiver. As a consequence, the overall throughput, which is the relevant QoS measure for elastic traffic, depends not only on the link capacities, but also on the number of concurrent flows. These have to be considered, when assigning bandwidth shares to elastic traffic classes.

The theory of processor sharing is applied to the dimensioning of networks for elastic traffic. In our context, processor sharing is interpreted as ideal bandwidth sharing among flows, whose arrival instances form a Poisson process and whose service times are generally distributed. The basic model, which has been proposed in the literature, is investigated and deficiencies are identified. In networks with longer round trip times the simple model underestimates the bandwidth needs. The reason for this is that TCP-specific features are not taken into account. Especially the slow-start mechanism of TCP delays the transfer process and, thus, decreases the actual throughput. Therefore, we propose an extension, which incorporates the slow-start behavior as well as the connection setup procedure and

the Nagle mechanism. The extended model adjusts the expected transaction times by taking into account estimates of the round trip time. This way, the required bandwidth can be determined more accurately.

The objective of the dimensioning process is to minimize the total link bandwidth in the network, while still being able to provide a certain average throughput for elastic traffic flows. We show how the processor sharing models, which in principle only relate to single links, can be applied to network dimensioning. The emphasis is on access networks and low-capacitated backbone links, as for these cases, capacities are most expensive. The validity of the processor sharing models is investigated through extensive packet-level simulations using the network simulator *ns2*. It can be demonstrated that for all relevant network scenarios, the formulae work sufficiently well. For longer round trip times the proposed extension significantly improves the accuracy of the processor sharing model.

Dimensioning for Stream Traffic

Stream traffic is related to applications and services with real-time critical demands. We specifically address session-oriented services such as Voice over IP, which require call admission control (CAC) in order to guarantee the desired QoS.

The objective of the dimensioning process is again the minimization of the total bandwidth in the network. Capacities need to be assigned in a way that call blocking probabilities do not exceed a certain threshold and that packet-level QoS of accepted flows is guaranteed throughout their duration. We assume that the latter is assured as long as a certain fixed bandwidth is provided to a flow on each link along its path.

Call admission control architectures for IP networks are still a matter of research, and numerous concepts have been proposed over the past years. Since their details of implementation are quite different, we derive abstract CAC models with similar consequences for network dimensioning. These are then used to develop appropriate dimensioning strategies. The decisive criterion of our abstraction is the location of *CA-controlled bandwidth components*, i.e., bandwidth components, which are considered for admission control. Every time a new call should be established, only the CA-controlled bandwidth components are checked whether they still provide enough free resources. Once a call is admitted, it might also traverse bandwidth shares, which are not subject to admission control (*non-CA-controlled* bandwidth shares). Based on this criterion, we distinguish between three main categories of CAC schemes: admission control at the ingress nodes only, at ingress and egress nodes, and at all nodes.

For each of the three categories, a two-step network dimensioning strategy is presented. At first, only the CA-controlled bandwidth components are dimensioned. Based on the reduced load approximation and the Erlang fixed point equations, a non-linear optimization problem is formulated and solved. The objective at this point is the minimization of the total CA-controlled bandwidth while the blocking probabilities are kept below the desired threshold (note that only CA-controlled bandwidth components introduce blocking). In the second step, the non-CA-controlled bandwidth shares are determined in a way that each of them is able to carry the worst-case traffic load. As these bandwidth shares are not subject to admission control, it has to be avoided that they become overloaded due to traversing flows,

which have been admission controlled elsewhere. The optimization problem of this step is solved through linear programming.

By applying the algorithms to several network scenarios and comparing the results for the various CAC schemes, the tradeoff between the amount of required bandwidth on the one hand, and the complexity of CAC mechanisms on the other can be demonstrated. Simple CAC schemes, which apply admission control only at the ingress node, need much higher bandwidths than more sophisticated ones, in order to achieve the same QoS guarantees. Nevertheless, for some network service providers, it might be justified to compensate the simplicity of a CAC mechanism with overdimensioning. This would be the case if the reduced management and administration effort equalizes the cost of additional bandwidth. Overall, the dimensioning procedures support providers in evaluating different CAC alternatives.

Outlook

The methodologies and algorithms proposed in this dissertation can serve as a basis for further research in the area of multi-service IP networking. In the following paragraphs a few ideas are selected and briefly discussed.

The presented routing optimization approach with conventional routing protocols could be extended in order to consider link and node failures ("resilience") as well as multiple traffic-load periods ("multi-period"). Instead of finding the optimum routing for one situation (normal load or no failures), a solution is searched for, which provides a tradeoff for several situations (multiple load periods or several failure states). A key issue of this type of optimization is certainly the definition of a meaningful objective function, which combines the QoS measures of the individual situations.

Using the hybrid genetic algorithm, resilience and multi-period optimization could be tackled by introducing additional iterations within the evaluation step of the algorithm (one loop for every failure or load scenario). This way, a genetic string would correspond to several routing scenarios, which have to be assessed with an appropriate fitness value. However, due to the additional loops, this approach could become too complex. Therefore, a different heuristic algorithm might have to be developed.

The dimensioning of networks for elastic traffic could also be further investigated. An interesting research issue would be the accuracy of the processor sharing model in large networks with high traffic aggregates. Especially flows that traverse many links with considerable ratios of cross-traffic might be negatively affected and, thus, might not achieve the predicted throughput. However, large-scale network scenarios require specialized simulators since packet-level simulations quickly become too complex for high traffic aggregates. A possible starting point could be a rate-based simulation approach as presented in [BB03], or even a hybrid packet-rate-based extension thereof. After all, it is necessary to find a compromise between simulation speed (as gained from rate-based simulation) and the required level of detail for TCP-specific features (as provided by *ns2*).

In the field of capacity assignment for stream traffic, the requirement of strict QoS provision under all circumstances could be softened. Currently, the dimensioning strategy assumes that packet-level QoS has to be guaranteed for all load situations, which are possible for a certain CAC dimensioning. As a consequence, some CAC schemes lead to rather high bandwidth

requirements. However, since certain worst-case load situations might be rather unlikely, it could be reasonable to allow temporary congestion on some links. As long as service degradation is highly improbable, it might be acceptable – especially if it leads to lower bandwidth sums.

So far, the algorithms for routing optimization and capacity assignment were considered separately because we believe that network service providers need modular planning methods rather than monolithic ones. However, during network planning an integrated approach is useful and an algorithm for routing optimization together with capacity assignment is desired. The hybrid genetic algorithm for routing optimization can easily be extended for network planning. This approach would in principle be the same as presented in [Rie98]. Within the genetic algorithm framework, routing and dimensioning can be performed sequentially. At first, all traffic flows are routed through the network taking into account the link metric values given by a genetic string. Then, the distribution of the traffic is determined and the links are dimensioned for stream as well as for elastic traffic classes [Val02]. A first implementation of this optimization strategy has returned encouraging results.

Finally, the presented algorithms need to be implemented in network planning tools in order to be usable by providers. However, as different providers require different algorithms, it is essential to have a modular tool architecture, which can be configured and extended as needed. In [Rie01] a generic framework for a highly flexible and distributed realization of a network planning tool is presented. Based on XML and CORBA, different suppliers could provide individual modules, which are then integrated into a comprehensive tool platform. The algorithms of this dissertation could be added to the distributed planning tool, which currently only exists as a prototype.

Appendix A

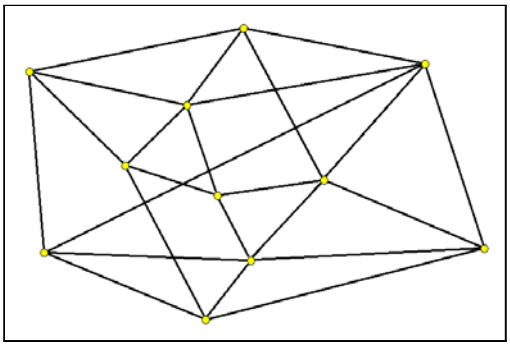
Sample Network Topologies

In this dissertation, five network scenarios were used for numerical evaluation of the presented algorithm: *N11*, *N20*, *N40*, *N50*, and *N100*.

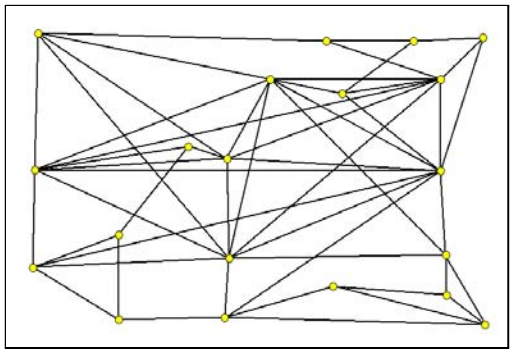
The topology of *N11* is a slightly modified version of the network specified by the COST 239 project [cos98]. It consists of 11 nodes and 48 unidirectional links. Network *N20* was published in [KB03] and contains 20 nodes and 102 links. Both networks, *N11* and *N20*, are flat networks without hierarchical structure. Scenario *N40* represents a network with two hierarchy levels. Each access node is connected to two backbone nodes. The network consists of 40 nodes and 150 links. Network *N50* is a hierarchical version of *N11*. Each backbone router connects three or four access nodes, giving a total of 50 nodes and 126 links. Finally, scenario *N100* represents a network where several access clusters are connected through a rather sparse backbone network. Furthermore, direct connections between neighboring access clusters exist. The network contains 100 nodes and 330 links.

	N11	**N20**	**N40**	**N50**	**N100**
number of nodes	11	20	40	50	100
number of links	48	102	150	126	330

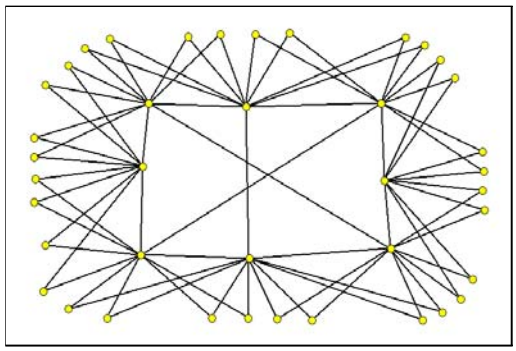
Table A.1: Characteristics of sample network scenarios



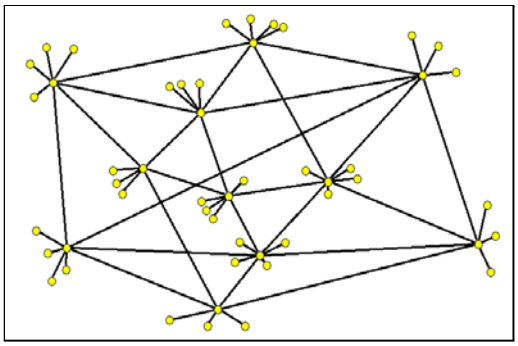
Network scenario *N11*



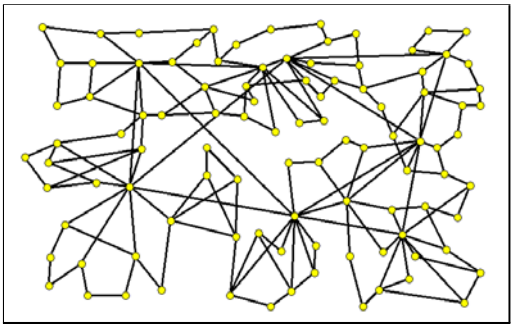
Scenario *N20*



Network scenario *N40*



Network scenario *N50*



Network scenario *N100*

Appendix B

Abbreviations

API	Application Programming Interface
AS	Autonomous System
ATM	Asynchronous Transfer Mode
BGP	Border Gateway Protocol
CAC	Call Admission Control
CAPEX	Capital Expenditure
CORBA	Common Object Request Broker Architecture
CR-LDP	Constraint-Routing LDP
DPE	Distributed Processing Environment
DSL	Digital Subscriber Line
DUAL	Diffusing Update Algorithm
ECMP	Equal-Cost Multi-Path
EGP	Exterior Gateway Protocol
EIGRP	Enhanced Interior Gateway Routing Protocol
ETSI	European Telecommunications Standards Institute
FEC	Forwarding Equivalence Class
GA	Genetic Algorithm
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communication
HGA	Hybrid Genetic Algorithm
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
IGRP	Interior Gateway Routing Protocol
IP	Internet Protocol
ISDN	Integrated Services Digital Network
IS-IS	Intermediate System – Intermediate System
ISO	International Organization for Standardization
ISP	Internet Service Provider
IT	Information Technology
ITU	International Telecommunications Union
LAN	Local Area Network
LDP	Label Distribution Protocol
LSF	Label-switched Flow (in MPLS)
LSP	Label-switched Path (in MPLS)

LSR	Label Switching Router
MC	Multicommodity
MPLS	Multiprotocol Label Switching
MSS	Maximum Segment Size
MTU	Maximum Transfer Unit
NGN	Next Generation Networking
OPEX	Operational Expenditure
OSPF	Open Shortest Path First
P2P	Peer-to-peer Networking
PS	Processor Sharing
PSTN	Public Switched Telephone Network
QoS	Quality of Service
RIP	Routing Information Protocol
RSVP	Resource Reservation Protocol
RSVP-TE	RSVP - Traffic Engineering
RTP	Real-time Transport Protocol
RTT	Round Trip Time
SIP	Session Initiation Protocol
SLA	Service Level Agreement
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VoIP	Voice over IP
VP	Virtual Path (in ATM)
VPN	Virtual Private Network
WFQ	Weighted Fair Queuing
WWW	World Wide Web
XML	Extensible Markup Language

Bibliography

[AAB00a] E. Altman, K. Avrachenkov, and C. Barakat. A Stochastic Model of TCP/IP with Stationary Random Losses. In *Proc. ACM SIGCOMM*, Stockholm, Sweden, August/September 2000.

[AAB00b] E. Altman, K. Avrachenkov, and C. Barakat. TCP in Presence of Bursty Losses. In *Proc. ACM SIGMETRICS*, Santa Clara, California, June 2000.

[AABD00] E. Altman, K. Avrachenkov, C. Barakat, and P. Dube. TCP over a Multi-State Markovian Path. In *Proc. International Conference on the Performance and QoS of Next Generation Networking (P&QNet)*, Nagoya, Japan, November 2000.

[ABG+01] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow. RSVP-TE: Extensions to RSVP for LSP Tunnels. RFC 3209, Internet Engineering Task Force, December 2001.

[ACE+02] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. Overview and Principles of Internet Traffic Engineering. RFC 3272, Internet Engineering Task Force, May 2002.

[ADF+01] L. Andersson, P. Doolan, N. Feldman, A. Fredette, and B. Thomas. LDP Specification. RFC 3036, Internet Engineering Task Force, January 2001.

[AGLAB94] R. Albrightson, J.J. Garcia-Luna-Aceves, and J. Boyle. EIGRP-A Fast Routing Protocol Based on Distance Vectors. In *Proceedings of Networld/Interop*, Las Vegas, USA, May 1994.

[AMA+99] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus. Requirements for Traffic Engineering Over MPLS. RFC 2702, Internet Engineering Task Force, September 1999.

[AMO93] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network Flows – Theory, Algorithms, and Applications*. Prentice Hall, Upper Saddle River, NJ, 1993.

[APS99] M. Allman, V. Paxson, and W. Stevens. TCP Congestion Control. RFC 2581, Internet Engineering Task Force, April 1999.

[BB03] M. Bahr and S. Butenweg. On rate-based Simulation of Communication Networks. In *Conference on Design, Analysis, and Simulation of Distributed Systems (DASD)*, Orlando, FL, USA, April 2003.

[BBC+98] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services. RFC 2475, Internet Engineering Task Force, December 1998.

[BBJ99] S. Borst, O. Boxma, and P. Jelenkovic. Generalized Processor Sharing with Long-Tailed Traffic Sources. In *Proceedings of the International Teletraffic Congress (ITC 16)*, Edinburgh, Scotland, 1999.

[BCS94] R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet Architecture: an Overview. RFC 1633, Internet Engineering Task Force, June 1994.

[Bel57] R.E. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, N.J., 1957.

[BF97] V. Brass and W.F. Fuhrmann. Traffic Engineering Experience from Operating Cellular Networks. *IEEE Communications Magazine*, 35(8):66–71, August 1997.

[BGH+02] J. Boyle, V. Gill, A. Hannan, D. Cooper, D. Awduche, B. Christian, and W.S. Lai. Applicability Statement for Traffic Engineering with MPLS. RFC 3346, Internet Engineering Task Force, August 2002.

[BGW98] A. Bley, M. Grötschel, and R. Wessäly. Design of Broadband Virtual Private Networks: Model and Heuristics for the B-WiN. Preprint SC 98-13, Konrad-Zuse-Zentrum für Informationstechnik, Berlin, March 1998.

[Bha99] R. Bhandari. *Survivable Networks – Algorithms for Diverse Routing*. Kluwer Academic Publishers, Boston, MA, 1999.

[BHKvdM01] J.V.L. Beckers, I. Hendrawan, R.E. Kooij, and R.D. van der Mei. Generalized Processor Sharing Performance Models for Internet Access Lines. In *Proc. of 9th IFIP Conference on Performance Modeling and Evaluation of ATM & IP Networks*, Budapest, Hungary, 2001.

[BK99] A. Berger and Y. Kogan. Multiclass Elastic Data Traffic: Bandwidth Engineering using Asymptotic Approximations. In *Proceedings of the International Teletraffic Congress (ITC 16)*, Edinburgh, Scotland, 1999.

[BK00] A.W. Berger and Y. Kogan. Dimensioning Bandwidth for Elastic Traffic in High-Speed Data Networks. *IEEE/ACM Transactions on Networking*, 8(5):643 – 654, October 2000.

[BLFF96] T. Berners-Lee, R. Fielding, and H. Frystyk. Hypertext Transfer Protocol – HTTP/1.0. RFC 1945, Internet Engineering Task Force, May 1996.

[BNC02] A. Broido, E. Nemeth, and KC Claffy. Internet Expansion, Refinement and Churn. *European Transactions on Telecommunications*, 1, January/February 2002.

[BOP94] Lawrence S. Brakmo, Sean W. O'Malley, and Larry L. Peterson. TCP vegas: New techniques for congestion detection and avoidance. In *ACM SIGCOMM*, pages 24–35, 1994.

[BPRR01] T. Bonald, A. Proutiére, G. Régnié, and J.W. Roberts. Insensitivity results in statistical bandwidth sharing. In *Proc. International Teletraffic Congress (ITC 17)*, 2001.

[BR00] T. Bonald and J.W. Roberts. Performance of Bandwidth Sharing Mechanisms for Service Differentiation in the Internet. In *ITC Specialist Seminar*, Monterey, California, September 2000.

[Bra89] R. Braden. Requirements for Internet Hosts – Communication Layers. RFC 1122, Internet Engineering Task Force, October 1989.

[BRF02] T. Bauschert, A. Riedl, and J. Frings. Planung von Multi-Service IP-Netzen. *Praxis der Informationsverarbeitung und Kommunikation (PIK)*, pages 90 – 97, February 2002.

[BZB+97] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification. RFC 2205, Internet Engineering Task Force, September 1997.

[Cah98] R.S. Cahn. *Wide Area Network Design – Concepts and Tools for Optimization*. Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1998.

[Cal90] R. Callon. Use of OSI IS-IS for Routing in TCP/IP and Dual Environments. RFC 1195, Internet Engineering Task Force, December 1990.

[CB92] B. Chinoy and H.-W. Braun. The National Science Foundation Network. Technical Report GA-21029, San Diego Supercomputer Center Applied Network Research Group, September 1992.

[CGR+00] F. Cuervo, N. Greene, A. Rayhan, C. Huitema, B. Rosen, and J. Segers. Megaco Protocol Version 1.0. RFC 3015, Internet Engineering Task Force, November 2000.

[Cha00] J. Charzinski. HTTP/TCP connection and flow characteristics. *Performance Evaluation*, 42:149–162, 2000.

[Cha02] J. Charzinski. Observed Performance of Elastic Internet Applications. *Computer Communication*, 26(8):914–925, 2002.

[Cla82] D. Clark. Window and Acknowledgment Strategy in TCP. RFC 813, Internet Engineering Task Force, July 1982.

[CNRS98] E. Crawley, R. Nair, B. Rajagopalan, and H. Sandick. A Framework for QoS-Routing in the Internet. RFC 2386, Internet Engineering Task Force, August 1998.

[Coc02] J.-Y. Cochhennec. Activities on Next-Generation Networks Under Global Information Infrastructure in ITU-T. *IEEE Communications Magazine*, 40(7), July 2002.

[Coh79] J.W. Cohen. The Multiple Phase Service Network with Generalized Processor Sharing. *Acta Informatica*, 12:245 – 284, 1979.

[Com00] D.E. Comer. *Interworking With TCP/IP: Principles, Protocols, and Architectures*. Prentice Hall, Upper Saddle River, NJ, 2000.

[cos98] COST 239: Ultra-high Capacity Optical Transmission Networks. http://barolo.ita.hsr.ch/cost239, 1998.

[DCB+02] B. Davie, A. Charny, J.C.R. Bennett, K. Benson, J.Y. Le Boudec, W. Courtney, S. Davari, V. Firoiu, and D. Stiliadis. An Expedited Forwarding PHB (Per-Hop Behavior). RFC 3246, Internet Engineering Task Force, March 2002.

[Ebe01] J. Eberspächer. Die Zukunft des Internet im Lichte von Konvergenz. In H. Kubicek, D. Klumpp, G. Fuchs, and A. Roßnagel, editors, *Internet@Future – Jahrbuch Telekommunikation und Gesellschaft 2001*, pages 17 – 27. Hüthig Verlag, Heidelberg, 2001.

[EGK+03] T. Engel, H. Granzer, B.F. Koch, M. Winter, P. Sampatakos, I.S. Venieris, H. Hussmann, F. Ricciato, and S. Salsano. AQUILA: Adaptive Resource Control for QoS Using an IP-Based Layered Architecture. *IEEE Communications Magazine*, 41(1):46–53, January 2003.

[ERP02] M. Ericsson, M.G.C. Resende, and P.M. Pardalos. A genetic algorithm for the weight setting problem in OSPF routing. *J. of Combinatorial Optimization*, 6:299–333, 2002.

[ETS03] ETSI. *ETSI Draft ES 202 915: Parlay 4.1*, March 2003.

[EVB01] J. Eberspächer, H.-J. Vögel, and C. Bettstetter. *GSM – Switching, Services and Protocols*. John Wiley & Sons, Ltd., New York, NY, 2 edition, 2001.

[Fel00] A. Feldmann. Characteristics of TCP Connection Arrivals. In K. Park and W. Willinger, editors, *Self-Similar Network Traffic And Performance Evaluation*. Wiley-Interscience Publication, New York, NY, 2000.

[FF62] L.R. Ford, Jr. and D.R. Fulkerson. *Flows in Networks*. Princeton University Press, Princeton, N.J., 1962.

[FF96] K. Fall and S. Floyd. Simulation-based Comparisons of Tahoe, Reno, and SACK TCP. *Computer Communication Review*, 26(3), July 1996.

[FGM+99] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616, Internet Engineering Task Force, June 1999.

[FH99] S. Floyd and T. Henderson. The NewReno Modification to TCP's Fast Recovery Algorithm. RFC 2582, Internet Engineering Task Force, April 1999.

[FK78] R.F. Farmer and I. Kaufmann. On the Numerical Evaluation of Some Basic Traffic Formulae. *Networks*, 8:153 – 186, 1978.

[Flo01] S. Floyd. A Report on Recent Developments in TCP Congestion Control. *IEEE Communications Magazine*, 39(4):84–90, April 2001.

[FML+03] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot. Packet-Level Traffic Measurements from the Sprint IP Backbone. *IEEE Network, to appear*, 2003.

[FRT02] B. Fortz, J. Rexford, and M. Thorup. Traffic Engineering with Traditional IP Routing Protocols. *IEEE Communications Magazine*, 40(10):118–124, October 2002.

[FT00] B. Fortz and M. Thorup. Internet Traffic Engineering by Optimizing OSPF Weights. In *Proceedings of IEEE INFOCOM*, pages 519–528, March 2000.

[FT02] B. Fortz and M. Thorup. Optimizing OSPF/IS-IS Weights in a Changing World. *IEEE Journal on Selected Areas in Communications (JSAC)*, 20(4):756–766, May 2002.

[FV00] K. Fall and K. Varadhan. *The ns Manual*. The VINT Project, 2000. http://www.isi.edu/nsnam/ns/doc-stable/index.html.

[GCR00] J. Glasmann, M. Czermin, and A. Riedl. Estimation of Token Bucket Parameters for Videoconferencing Systems in Corporate Networks. In *Proceedings of 8th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Split, Croatia, October 2000.

[GDF00] I. Guardini, P. D'Urso, and P. Fasano. The Role of Internet Technology in Future Mobile Data Systems. *IEEE Communications Magazine*, 38(11):68–72, November 2000.

[Gir90] A. Girard. *Routing and Dimensioning in Circuit-Switched Networks*. Addison-Wesley, Reading, MA, USA, 1990.

[GLA93] J.J. Garcia-Lunes-Aceves. Loop-Free Routing Using Diffusing Computations. *IEEE/ACM Transactions on Networking*, 1(1):130–141, February 1993.

[GM02] J. Glasmann and H. Müller. Resource Management Architecture for Realtime Traffic in Intranets. In *Proceedings of Networks 2002, Joint IEEE International Conferences ICN and ICWLHN*, Atlanta, USA, 2002.

[GMW93] P.E. Gill, W. Murray, and M.H. Wright. *PracticalOptimization*. Academic Press, 1993.

[Gol89] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Massachusetts, 1989.

[GP99] R. Guérin and V. Peris. Quality-of-service in packet networks: basic mechanisms and directions. *Computer Networks*, 31(3):169–189, 1999.

[GRR00] N. Greene, M. Ramalho, and B. Rosen. Media Gateway Control Protocol Architecture and Requirements. RFC 2805, Internet Engineering Task Force, April 2000.

[GSE+00] R.J. Gibbens, S.K. Sargood, C. Van Eijl, F.P. Kelly, H. Azmoodeh, R.H Macfadyen, and N.W. Macfadyen. Fixed-Point Models for the End-to-End Performance Analysis of IP Networks. In *13th ITC Specialist Seminar: IP Traffic Measurement, Modeling and Management*, Monterey, California, September 2000.

[HB96] J. Hawkinson and T. Bates. Guidelines for creation, selection, and registration of an Autonomous System (AS). RFC 1930, Internet Engineering Task Force, March 1996.

[HBWW99] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. Assured Forwarding PHB Group. RFC 2597, Internet Engineering Task Force, June 1999.

[Hed88] C. Hedrick. Routing Information Protocol. RFC 1058, Internet Engineering Task Force, June 1988.

[HLN97] D.P. Heyman, T.V. Lakshman, and A.L. Neidhardt. A New Method for Analysing Feedback-Based Protocols with Applications to Engineering Web Traffic over the Internet. In *Proc. of 1997 ACM SIGMETRICS*, pages 24 – 38, Seattle, Washington, 1997.

[Hol75] J.H. Holland. Adaptation in natural and artifical systems. Technical report, Ann Arbor: The University of Michigan Press, 1975.

[HS03] G. Haßlinger and S. Schnitter. Optimized Traffic Load Distribution in MPLS Networks. In G. Anandalingam and S. Raghavan, editors, *Telecommunications Network Design and Management*. Kluwer Academic Publishers, Boston, 2003.

[Hui00] Christian Huitema. *Routing in the Internet*. Prentice-Hall, Inc., Upper Saddle River, N.J., 2 edition, 2000.

[Hus99] G. Huston. *ISP Survival Guide - Strategies for Running a Competitive ISP*. John Wiley & Sons, Inc., New York, NY, 1999.

[HWB93] K. Claffy H.-W. Braun. Network analysis issues for a public Internet. Technical Report GA-21350, San Diego Supercomputer Center, May 1993.

[HY01a] K. Holmberg and D. Yuan. Optimal Network Design and Routing for IP Traffic. In *Third International Workshop on Design of Reliable Communication Networks (DRCN 2001)*, Budapest, Hungary, October 2001.

[HY01b] K. Holmberg and D. Yuan. Optimization of Internet Protocol Network Design and Routing. Research Report LiTH-MAT-R-2001-07, Department of Mathematics, Linkoping Institute of Technology, Sweden, March 2001.

[Int03] International Packet Communications Consortium (IPCC), San Ramon, CA, USA. *Packet Communications Reference Architecture - Version 2.0*, April 2003.

[ITU93] ITU-T, Geneva, Switzerland. *ITU-T Recommendation E.800: Terms and Definitions Related to Quality of Service and Network Performance Including Dependability*, August 1993.

[ITU99] ITU-T, Geneva, Switzerland. *ITU-T Recommendation G.108: Application of the E-model: A Planning Guide*, September 1999.

[ITU00a] ITU-T, Geneva, Switzerland. *ITU-T Recommendation H.225.0: Call signalling protocols and media stream packetization for packet-based multimedia communication systems*, November 2000.

[ITU00b] ITU-T, Geneva, Switzerland. *ITU-T Recommendation H.323: Packet-based multimedia communications systems*, November 2000.

[Jac88] V. Jacobson. Congestion Avoidance and Control. *ACM Computer Communication Review; Proceedings of the Sigcomm '88 Symposium in Stanford, CA, August, 1988*, 18, 4:314–329, 1988.

[JAC+02] B. Jamoussi, L. Andersson, R. Callon, R. Dantu, L. Wu, P. Doolan, T. Worster, N. Feldman, A. Fredette, M. Girish, E. Gray, J. Heinanen, T. Kilty, and A. Malis. Constraint-Based LSP Setup using LDP. RFC 3212, Internet Engineering Task Force, January 2002.

[JTG00] J. de Keijzer, D. Tait, and R. Goedman. JAIN: A New Approach to Services in Communication Networks. *IEEE CommunicationsMagazine*, 38(1):94 – 99, January 2000.

[KB03] S. Köhler and A. Binzenhöfer. MPLS Traffic Engineering in OSPF Networks – a combined Approach. Technical Report 304, University of Würzburg, February 2003.

[Ker93] A. Kershenbaum. *Telecommunications Network Design Algorithms*. McGraw-Hill, Inc., New York, NY, 1993.

[KJC+03] Y. Kim, B.J. Jeong, J. Chung, C.-S. Hwang, J.S. Ryu, K.-H. Kim, and Y.K. Kim. Beyond 3G: Vision, Requirements, and Enabling Technologies. *IEEE Communications Magazine*, 41(3):120–124, March 2003.

[KK00] A. Kherani and A. Kumar. Performance Analysis of TCP with Nonpersistent Sessions. In *Workshop on Modeling of Flow and Congestion Control*, September 2000.

[KKS01] V. Kumar, M. Korpi, and S. Sengodan. *IP Telephony with H.323*. John Wiley & Sons, Inc., New York, NY, 2001.

[Kle76] L. Kleinrock. *Queueing Systems Volume II: Computer Applications*. Wiley-Interscience Publication, New York, 1976.

[KO02] L. Krank and H. Orlamünder. Future Telecommunication Traffic - A Methodology for Estimation. In *Proceedings of 10th International Telecommunication Network Strategy and Planning Symposium (NETWORKS 2002)*, pages 139 – 144, Munich, Germany, June 2002.

[KSK02] S. Köhler, D. Staehle, and U. Kohlhaas. Optimization of IP Routing by Link Cost Specification. In *Internet Traffic Engineering and Traffic Management, 15-th ITC Specialist Seminar*, Wuerzburg Germany, July 2002.

[LF03] H.-L. Lu and I. Faynberg. An architectural framework for support of quality of service in packet networks. *IEEE Communications Magazine*, 41(6):98 – 105, June 2003.

[LHJ+00] B. Li, M. Hamdi, D. Jiang, X.-R. Cao, and Y.T. Hou. QoS-Enabled Voice Support in the Next-Generation Internet: Issues, Existing Approaches and Challenges. *IEEE Communications Magazine*, 38(4):54–61, January 2000.

[Lin98] K. Lindberger. Delays in ABR-like traffic and relations to processor sharing. COST Technical Document COST 257TD(98)10, 1998.

[Lin99] K. Lindberger. Balancing Quality of Service, Pricing and Utilisation in Multiservice Networks with Stream and Elastic Traffic. In *Proceedings of the International Teletraffic Congress (ITC 16)*, Edinburgh, Scotland, 1999.

[LM97] T.V. Lakshman and P. Madhow. The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss. *IEEE/ACM Transactions on Networking*, 5(3):336 – 350, June 1997.

[LMS97] T.V. Lakshman, U. Madhow, and B. Suter. Window-based error recovery and flow control with a slow acknowledgement channel: a study of TCP/IP performance. In *Proc. IEEE Infocom*, 1997.

[LMS00] T.V. Lakshman, P. Madhow, and B. Suter. TCP/IP Performance with Random Loss and Bidirectional Congestion. *IEEE/ACM Transactions on Networking*, 8(5):541 – 555, October 2000.

[LPHC02] Y.-B. Lin, A.-C. Pang, Y.-R. Haung, and I. Chlamtac. An All-IP Approach for UMTS Third-Generation Mobile Networks. *IEEE Network*, 16(5):8–19, September/October 2002.

[Mal94] G. Malkin. RIP Version 2 - Carrying Additional Information. RFC 1723, Internet Engineering Task Force, November 1994.

[Mal97] G. Malkin. RIPng for IPv6. RFC 2080, Internet Engineering Task Force, January 1997.

[MBD00] S. Meddus, G. Bruce, and S. Davis. Opening Up Networks with JAIN Parlay. *IEEE CommunicationsMagazine*, 38(4):136 – 143, April 2000.

[McC98] J.D. McCabe. *Practical Computer Network Analysis and Design.* Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1998.

[MH00] A. Mena and J. Heidemann. An Empirical Study of Real Audio Traffic. In *Proceedings of the IEEE Infocom*, pages 101–110, Tel-Aviv, Israel, March 2000. IEEE.

[Mil84] D.L. Mills. Exterior Gateway Protocol formal specification. RFC 904, Internet Engineering Task Force, April 1984.

[MK02a] E. Mulyana and U. Killat. A Hybrid Genetic Algorithm Approach for OSPF Weight Setting Problem. In *Proceedings of the 2nd Polish-German Teletraffic Symposium PGTS 2002*, Gdansk Poland, September 2002.

[MK02b] E. Mulyana and U. Killat. An Alternative Genetic Algorithm to Optimize OSPF Weights. In *Internet Traffic Engineering and Traffic Management, 15-th ITC Specialist Seminar*, Wuerzburg Germany, July 2002.

[MKSB02] J. Milbrandt, S. Köhler, D. Staehle, and L. Berry. Decomposition of Large IP Networks for Routing Optimization. Technical Report 293, University of Würzburg, February 2002.

[MM00] A.R. Modarressi and S. Mohan. Control and Management in Next-Generation Network: Challenges and Opportunities. *IEEE Communication-sMagazine*, 38(10):94 – 102, October 2000.

[MMFR96] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP Selective Acknowledgment Options. RFC 2018, Internet Engineering Task Force, October 1996.

[MO99] Archan Misra and Teunis J. Ott. The Window Distribution of Idealized TCP Congestion Avoidance with Variable Packet Loss. In *INFOCOM (3)*, pages 1564–1572, 1999.

[Moy98] J. Moy. OSPF Version 2. RFC 2328, Internet Engineering Task Force, April 1998.

[MR99] L. Massoulié and J. Roberts. Arguments in favour of admission control for TCP flows. In *Proceedings of the International Teletraffic Congress (ITC 16)*, Edinburgh, Scotland, 1999.

[MSM97] M. Mathis, J. Semke, and J. Mahdavi. The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm. *Computer Communications Review*, 27(3), 1997.

[Nag84] J. Nagle. Congestion Control in IP/TCP Internetworks. RFC 896, Internet Engineering Task Force, January 1984.

[NW99] J. Nocedal and S.J. Wright. *Numerical Optimization.* Springer, 1999.

[OKM96] T. Ott, J. Kemperman, and M. Mathis. The stationary behavior of ideal TCP congestion avoidance, 1996. ftp://ftp.bellcore.com/pub/tjo/TCPwindow.ps.

[OR00] P. Olivier and J. Roberts. IP link dimensioning in the context of DiffServ. COST Technical Document COST 257TD(00)01, 2000.

[Ott84] T.J. Ott. The Sojourn-time Distribution in the M/G/1 Queue with Processor Sharing. *J. Appl. Prob.*, 21:360 – 378, 1984.

[PD00] L.L. Peterson and B.S. Davie. *Computer Networks - A Systems Approach.* Morgan Kaufmann Publishers, San Francisco, 2 edition, 2000.

[PF01] J. Padhye and S. Floyd. On Inferring TCP Behavior. In *ACM SIGCOMM*, August 2001.

[PFTK98] J. Padhye, V. Firoiu, D. Towsley, and J. Krusoe. Modeling TCP Throughput: A Simple Model and its Empirical Validation. In *ACM SIGCOMM '98 conference on Applications, technologies, architectures, and protocols for computer communication*, pages 303–314, Vancouver, CA, 1998.

[PFTK00] J. Padhye, V. Firoiu, D.F. Towsley, and J.F. Kurose. Modeling TCP Reno Performance: A Simple Model and Its Empirical Validation. *IEEE/ACM Transactions on Networking*, 8(2):133 – 145, April 2000.

[Pos81] J. Postel. Transmission Control Protocol. RFC 793, Internet Engineering Task Force, September 1981.

[Pos83] J. Postel. The TCP Maximum Segment Size and Related Topics. RFC 879, Internet Engineering Task Force, November 1983.

[QvdBM99a] R. Núñez Queija, J.L. van den Berg, and M.R.H. Mandjes. Performance evaluation of strategies for integration of elastic and stream traffic. In *Proceedings of the International Teletraffic Congress (ITC 16)*, Edinburgh, Scotland, 1999.

[QvdBM99b] R. Núñez Queija, J.L. van den Berg, and M.R.H. Mandjes. Performance evaluation of strategies for integration of elastic and stream traffic. Research Report – Probability, Networks and Algorithms PNA-R9903, Centrum voor Wiskunde en Informatica, February 1999.

[RBF02] A. Riedl, T. Bauschert, and J. Frings. A Framework for Multi-Service IP Network Planning. In *Proceedings of 10th International Telecommunication Network Strategy and Planning Symposium (Networks 2002)*, Munich, Germany, June 2002.

[RBF03] A. Riedl, T. Bauschert, and J. Frings. On the Dimensioning of Voice over IP Networks for various Call Admission Control Schemes. In *to be presented at 18th International Teletraffic Congress (ITC 18)*, Berlin, Germany, August 2003.

[Res02] M.G.C. Resende. A memetic algorithm for OSPF routing. presented at the Sixth INFORMS Telecom Conference, March 2002.

[REV01] M. Roughan, A. Erramilli, and D. Veitch. Network Performance for TCP Networks – Part I: Persistent Sources. In *Proc. International Teletraffic Congress (ITC 17)*, 2001.

[RG02] A. Riedl and J. Glasmann. On the Design of Resource Management Domains. In *Proc. of 2002 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS 2002)*, San Diego, CA, USA, July 2002.

[Rie98] A. Riedl. A Versatile Genetic Algorithm for Network Planning. In *Proceedings of Open European Summer School on Network Management and Operation (EUNICE'98)*, pages 97 – 103, Munich, Germany, August/September 1998.

[Rie01] A. Riedl. A CORBA/XML-based Architecture for Distributed Network Planning Tools. In *Proc. of International Conference on Enterprise Information System (ICEIS)*, Setúbal, Portugal, July 2001.

[Rie02] A. Riedl. A Hybrid Genetic Algorithm for Routing Optimization in IP Networks Utilizing Bandwidth and Delay Metrics. In *Proceedings of IEEE Workshop on IP Operations and Management (IPOM)*, Dallas, USA, October 2002.

[Rie03] A. Riedl. Optimized Routing Adaptation in IP Networks Utilizing OSPF and MPLS. In *Proceedings of IEEE International Conference on Communications (ICC)*, Anchorage, USA, May 2003.

[RL95] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP-4). RFC 1771, Internet Engineering Task Force, March 1995.

[RLGPC+99] A. Reyes-Lecuona, E. González-Parada, E. Casilari, J.C. Casasola, and A. Díaz-Estrella. A page-oriented WWW traffic model for wireless system simulations. In *Proceedings of the International Teletraffic Congress (ITC 16)*, Edinburgh, Scotland, 1999.

[RM98] J. Roberts and L. Massoulié. Bandwidth sharing and admission control for elastic traffic. In *ITC Specialist Seminar*, Yokohama, Japan, October 1998.

[Rob97] J. Roberts. Realizing quality of service guarantees in multiservice networks. In *Proceedings of the IFIP Conference PMCCN'97*, Tsukuba Science City, Japan, November 1997.

[Rob98] T.G. Robertazzi. *Planning Telecommunication Networks*. IEEE Press, New York, NY, 1998.

[RPBP00a] A. Riedl, M. Perske, T. Bauschert, and A. Probst. Dimensioning of IP Access Networks with Elastic Traffic. In *Proceedings of 9th International Telecommunication Network Planning Symposium (Networks 2000)*, Toronto, Canada, September 2000.

[RPBP00b] A. Riedl, M. Perske, T. Bauschert, and A. Probst. Investigation of the M/G/R Processor Sharing Model for Dimensioning of IP Access Networks with Elastic Traffic. In *First Polish-German Teletraffic Symposium (PGTS)*, Dresden, Germany, September 2000.

[RR01] K.G. Ramakrishnan and M.A. Rodrigues. Optimal Routing in Shortest-Path Data Networks. *Bell Labs Technical Journal*, pages 117–138, January–June 2001.

[RS02] A. Riedl and D.A. Schupke. A Flow-Based Approach for IP Traffic Engineering Utilizing Routing Protocols With Multiple Metric Types. presented at the Sixth INFORMS Telecom Conference, March 2002.

[RSC+02] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261, Internet Engineering Task Force, June 2002.

[RVC01] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol Label Switching Architecture. RFC 3031, Internet Engineering Task Force, January 2001.

[Sch02] R. Schollmeier. Peer-to-Peer Networking. Applications for and Impacts on Future IP-Based Networks. In *3. ITG Fachtagung Netze und Anwendungen: Neue Kommunikationsanwendungen in modernen Netzen*, Duisburg, Germany, February 2002.

[SGG02] S. Saroiu, P. Krishna Gummadi, and S.D. Gribble. A Measurement Study of Peer-to-Peer File Sharing Systems. In *Proceedings of Multimedia Computing and Networking 2002 (MMCN '02)*, San Jose, CA, USA, January 2002.

[SH02] S. Schnitter and G. Haßlinger. Heuristic Solutions to the LSP-Design for MPLS. In *Proceedings of 10th International Telecommunication Network Strategy and Planning Symposium (NETWORKS 2002)*, pages 269–273, Munich, Germany, June 2002.

[SKK00a] D.C. Schmidt, V. Kachroo, and F. Kuhns. Developing Next-Generation Distributed Applications with QoS-Enabled DPE Middleware. *IEEE CommunicationsMagazine*, 38(10):112 – 123, October 2000.

[SKK00b] D. Staehle, S. Köhler, and U. Kohlhaas. Towards an optimization of the routing parameters for IP networks. Technical Report 258, University of Würzburg, May 2000.

[SKO+02] K. Stordahl, K.O. Kalhagen, B.T. Olsen, J. Lydersen, B. Olufsen, and N.K. Elnegaard. Traffic forecast models for the transport network. In *Proceedings of 10th International Telecommunication Network Strategy and Planning Symposium (NETWORKS 2002)*, pages 145 – 150, Munich, Germany, June 2002.

[SP98] O. Schelén and S. Pink. Resource Reservation Agents in the Internet. *Journal of High Speed Networks, Special issue on Multimedia Networking*, 1998.

[SRGT03] S. Sharafeddine, A. Riedl, J. Glasmann, and J. Totzke. On Traffic Characteristics and Bandwidth Requirements of Voice over IP Applications. In *Proceedings of*, Antalya, Turkey, June/July 2003.

[Ste94] W. R. Stevens. *TCP/IP Illustrated, Volume 1: The Protocols*. Addison-Wesley, Reading, MA, 1994.

[SW02] S. Sen and J. Wang. Analyzing peer-to-peer traffic across large networks. In *Proceedings of Second Annual ACM Internet Measurement Workshop*, pages 137–150, November 2002.

[TAP+01] P. Trimintzios, I. Andrikopoulos, G. Pavlou, P. Flegkas, D. Griffin, P. Georgatsos, D. Goderis, Y. T'Joens, L. Georgiadis, C. Jacquenet, and R. Egan. A Management and Control Architecture for Providing IP Differentiated Services in MPLS-Based Networks. *IEEE Communications Magazine*, 39(5), May 2001.

[TB00] D. Tang and M. Baker. Analysis of a Local-Area Wireless Network. In *Proc. of The Sixth Annual International Conference on Mobile Computing and Networking (MobiCom 2000)*, Bosten, MA, USA, August 2000.

[TB02] D. Tang and M. Baker. Analysis of a Metropolitan-Area Wireless Network. *Wireless Networks*, 8:107–120, 2002.

[TMW97] K. Thompson, G.J. Miller, and R. Wilder. Wide-Area Internet Traffic Patterns and Characteristics. *IEEE Network*, 11(6):10 – 23, November/December 1997.

[Val02] G. Valavanis. An Object-Oriented Tool for Dimensioning QoS-enabled IP Networks. Master's Thesis, Lehrstuhl für Kommunikationsnetze, Technische Universität München, September 2002.

[vdBvdMG+01] H. van den Berg, R. van der Mei, B. Gijsen, M. Pikaart, and R. Vranken. Processing times in transaction servers with Quality of Service differentiation. In *11th GI/ITG Conference on Measuring, Modelling and Evaluation of Computer and Communication Systems (MMB)*, Aachen, Germany, September 2001.

[vdMSK02] J. van der Merwe, S. Sen, and C. Kalmanek. Streaming Video Traffic: Characterization and Network Impact. In *Proc. 7th International Workshop on Web Content Caching and Distribution (WCW)*, Boulder, CO, USA, August 2002.

[VLLX02] J. De Vriendt, P. Lainé, C. Lerouge, and X. Xu. Mobile Network Evolution: A Revolution on the Move. *IEEE Communications Magazine*, 40(4):104–111, April 2002.

[WC92] Z. Wang and J. Crowcroft. Analysis of Shortest-Path Routing Algorithms in a Dynamic Network Environment. *Computer Communication Review*, 22(2):63–71, 1992.

[WW99] Y. Wang and Z. Wang. Explicit routing algorithms for Internet traffic engineering. In *Proceedings of the 8th International Conference on Computer Communications and Networks (ICCCN 1999)*, pages 582–588, October 1999.

[WWZ01] Y. Wang, Z. Wang, and L. Zhang. Internet Traffic Engineering without Full Mesh Overlaying. In *Proceedings of IEEE INFOCOM*, pages 565–571, April 2001.

[XHBN00] X. Xiao, A. Hannan, B. Bailey, and L.M. Ni. Traffic Engineering with MPLS in the Internet. *IEEE Network Magazine*, pages 28–33, March/April 2000.

[Yas87] S.F. Yashkov. Processor-Sharing Queues: Some Progress in Analysis. *Queueing Systems*, 2:1 – 17, 1987.

[Yas92] S.F. Yashkov. Mathematical Problems in the Theory of Shared-Processor Systems. *J. Soviet Mathematics*, 58:101 – 147, 1992.

[ZB98] A.P. Zwart and O.J. Boxma. Sojourn time asymptotics in the M/G/1 processor sharing queue. Technical Report PNA-R9802, Centrum voor Wiskunde en Informatica (CWI), April 1998.

[Zha95] H. Zhang. Service disciplines for guaranteed performance service in packet-switching networks. *Proceedings of the IEEE*, 83(10):1374 – 1396, October 1995.

[Zin02] A. Zinin. *Cisco IP Routing.* Addison-Wesley, Boston, MA, 2002.

[ZVTZ02] T.B. Zahariadis, K.G. Vaxevanakis, C.P. Tsantilas, and N.A. Zervos. Global Roaming in Next-Generation Networks. *IEEE Communications Magazine*, 40(2):145–151, February 2002.