Lehrstuhl für Kommunikationsnetze

IP Network Planning for Realtime Services with Statistical QoS Guarantees

Sanaa Sharafeddine

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor–Ingenieurs

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr.-Ing. Fernando Puente León
Prüfer der Dissertation:	
	1. Univ.-Prof. Dr.-Ing. Jörg Eberspächer
	2. Univ.-Prof. Dr. rer. nat. Paul Müller, Technische Universität Kaiserslautern

Die Dissertation wurde am 26.01.2005 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 04.06.2005 angenommen.

Preface

This dissertation was written during my time as a research assistant at the Institute of Communication Networks (LKN) at Munich University of Technology (TUM). This dissertation would not have been possible without the assistance and cooperation of several people who have shaped this work in various ways.

First, I would like to express my profound appreciation to my PhD advisor Prof. Dr.-Ing. Jörg Eberspächer for his continued generous support. I am thankful to him for being available whenever I needed his advice and for offering me a working place at his institute. There, I was given full access to the institute's facilities and given the chance to collaborate with several remarkable colleagues. Thanks also to Prof. Dr. Paul Müller for accepting to serve as a second examiner for this thesis.

I wish to acknowledge the enjoyable atmosphere at TUM which is definitely due to the LKN staff and one LNT member. To all I am especially grateful. However, I would like to note some people who stand out most notably in my mind as contributing to the success of this work. To Prof. Dr.-Ing. Zaher Dawy, who tirelessly does his best to improve on my best, thank you! The many hours we spent chatting and the "weekend workshops" we organized have directly contributed to the ideas in this dissertation. To Prof. Dr.-Ing. Anton Riedl, who was the first colleague to meet at LKN and who has since then contributed to the advancement of this work, I highly appreciate it. We have had extraordinarily fine technical conversations that were a valuable source of inspiration to me. I am additionally thankful to him for his careful proofreading for major parts of this dissertation. To Andrea Bör, you made my stay at LKN so enjoyable. Thanks for encouraging me to move my working place to LKN. To Dr.-Ing. Thomas Bauschert and Dr.-Ing. Josef Glasmann, thanks for sharing your experience that helped me during different phases of this work. To Dr.-Ing. Martin Maier, many thanks for all the technical support at LKN.

This work has been sponsored by Siemens AG in Munich. Special thanks are dedicated to Mr. Eberhard Wildgrube and to Dr.-Ing. Harald Müller for initiating this project and awarding me a scholarship to continue my studies. I am also thankful to Mr. Jürgen Totzke and Mr. Alfons Fartmann for the fruitful joint cooperation. Additional thanks are dedicated to Mr. Totzke for the many stimulating discussions that gave me the practical aspect of my work and for proofreading parts of this dissertation.

Last but definitely most importantly, a deep heartfelt thank you goes to those who endlessly sacrifice for my well-being, my wonderful father and mother. Without their infinite support and devotion, this work would never have been. I am extremely lucky to have such a loving family: my sisters Aziza and Sahar and my brothers Ahmad, Ali, and Samer. I am also lucky to have such an affectionate and dedicated second half, my husband Zaher. He makes our life a wonderful journey.

München, June 2005 Sanaa Sharafeddine

To Mom and Dad
and
To Zaher

Abstract

This dissertation investigates various areas of IP network planning for realtime services. These areas are traffic characterization, capacity assignment, robust network dimensioning, and network planning tool development. In regards to traffic characterization, various clients of realtime services are tested in different situations and their traffic is characterized using known models. Motivated by the fact that hard quality guarantees incur extremely high costs, this dissertation exploits the issue of statistical guarantees and introduces two novel approaches for capacity assignment for interactive voice and video services, respectively. Subsequently, it proposes a statistical way that accounts for traffic demand variability in the context of network dimensioning to assure a sufficiently robust network. Finally, a software planning tool with a generic layered architecture is developed for converged IP networks.

Zusammenfassung

Diese Arbeit befaßt sich mit verschiedenen Aspekten der Planung von IP-Netzen für echtzeitkritische Dienste. Es werden primär die Themenbereiche Verkehrscharakterisierung und Kapazitätsplanung unter Berücksichtigung von Dienstgüte- und Robustheitsanforderungen adressiert. Im Rahmen der Untersuchungen zur Verkehrscharakterisierung sind verschiedene Echtzeit-Applikationen in unterschiedlichen Anwendungsszenarien betrachtet worden. Als Ergebnis der Forschungsarbeiten zur Kapazitätsplanung werden zwei neue Ansätze vorgestellt, welche sich besonders für IP-Netze mit echtzeitkritischen interaktiven Sprach- und Videodiensten eignen. Bei den hier vorgeschlagenen neuen Verfahren wird ein statistisches Kriterium zur Einhaltung der Dienstgüte verwendet. Je nach Robustheitsanforderungen der Netzbetreiber sind somit große Kapazitäts- bzw. Kosteneinsparungen realisierbar. Schließlich ist bereits ein Planungstool für konvergente IP-Netze mit Multimedia-Diensten entwickelt worden.

Contents

1

Introduction

During the tech bubble in the late 1990s, substantial investments were put in the telecom industry that amounted to around three quarters of a trillion dollars [Ins04]. Since the market crash of the year 2000, layoffs, bankruptcies, and even accounting scandals have been witnessed during three rugged years in which company fortunes have been dramatically pushed downward. It is only until the year 2004 that the telecommunications industry started to recover. It is expected that billions of dollars will be spent over the next years on a new generation of networks integrating voice, video, and data services rather than on mere maintenance and modest upgrades to existing networks. The ultimate goal is to blur the lines between the different existing networks leading to a completely unified network.

1.1 Convergence of Networks

Network convergence is defined as the ability of all networks to seamlessly carry all services. Traditionally, separate networks existed for different services such as voice, data, telex, and television. Each network type was optimized to the attributes of its associated application or service, making it rather inapt of carrying other traffic. For example, telephone networks, fixed and mobile, still cannot compete with broadcasting networks for the bulk transmission of high quality moving pictures. This fact is due to the inherent characteristics of each of the two network types. Traditional telephone providers, cable TV system operators, and internet service providers were once struggling for their survival and competing to create the converged network using their own infrastructure. However, it has become a general consensus today that telecommunications services are moving slowly towards purely IP-based networks that will do everything, thus, forming all-IP converged networks.

Conventionally, there have been two general communications paradigms, namely the telecommunications paradigm and the computer networks paradigm. The former refers to a small number of services, which usually have a persistent nature. The telecommunications paradigm is characterized by its constant bit rate (in traditional networks), realtime, and peer-to-peer

features. This paradigm is not in anyway adapted to highly bursty bit rate traffic, best-effort transactions, and client/server mode of operation. These features, on the other hand, characterize the communication requirements of computer networks. The differences between the two paradigms are even magnified when it comes to telecommunications-computer networks convergence. A unified network is generally intended to carry both service categories. For such convergence, several approaches are possible and are worth of being investigated [XN99]. Among all approaches, the internet protocol (IP) is widely accepted as the basis for converged next generation networks. As a result, one of the main steps towards convergence is to upgrade current IP-based computer networks (e.g. the internet) to support telecom services leading to a multiservice network architecture.

Towards achieving a global service convergence, a transfer system capable of carrying different flow types with different performance criteria is necessary. The converged network is required to support non-persistent, persistent, client/server, and peer-to-peer services. Persistent services are services where the session is not terminated with temporary user activity. A session remains open as long as the end parties are exchanging information although one user might stay inactive for a long period of time. To operate, a persistent service requires a control plane to be setup and signaling mechanisms established between the control plane entities. A control plane is defined as the set of entities responsible for the setup, modification, and release of a persistent service instance. In conventional telecommunications networks, examples of control architectures are the intelligent network architecture, the Parlay architecture, and the programmable network architecture. In converged networks, examples of control architectures are the H.323 architecture, the SIP architecture, and the soft-switches with the H.248 architecture. In the client/server mode, there is no context association between the client and the server. The server runs indefinitely waiting for queries initiated by clients and answering these queries. The client and the server operate asynchronously. In the peer-to-peer mode, both entities act cooperatively with context sharing forming a single global context for one service instance.

1.2 Quality of Service

With the evolution of the internet protocol as the ubiquitous infrastructure, various categories of services are demanded. One category provides internet services where customers are willing to pay a certain price to make their service reliable and fast enough. Within this category, a couple of service classes can be provided in decreasing quality such as gold, silver, and bronze. A second category can provide timely service with low delay and low jitter (inter-packet delay variation). Examples of applications belonging to this service category are telephony and videoconferencing. For such a service, customers are willing to pay a higher price to achieve premium communications quality. Finally, the best-effort service is granted to customers who are not willing to pay any additional price for improved quality.

IP networks were originally designed for best effort service that grants no guarantees to timeliness and reliability of delivery. Traffic is processed as soon as possible regardless of its type and performance criteria. However, in order to be widely acceptable as the universal infrastructure for all types of services, IP networks have to support traffic generated by any of the new evolving applications. This places a huge burden on IP networks, which have to handle various services with different traffic characteristics and quality of service (QoS) demands.

It is sometimes argued that mechanisms are not necessary to provide QoS for the different traffic classes if bandwidth becomes excessive and packet processing/forwarding within routers

is quick. These two conditions, if available, can result in negligible packet delay and consequently premium QoS. Fibers and wavelength division multiplexing (WDM) will contribute in achieving one of the conditions by making bandwidth cheap and abundant. However, the advent of high-speed links is causing more heterogeneity in computer networks making them more challenging to control. The evolution of computer networks in the previous years shows that regardless of how much bandwidth is available, there will always be new applications that consume them. Furthermore, even huge link capacities cannot guarantee a faultless service with no phases of interruption and quality degradation. Therefore, mechanisms are still needed that provide QoS and importantly differentiated QoS, where traffic classes are not treated equally but differently. This fact drives the major router/switch vendors, such as Cisco, Entrasys, Extreme, 3Com, and Huawei, to provide QoS mechanisms in their products.

The Internet Engineering Task Force (IETF) has proposed several mechanisms and approaches as an extension to the traditional internet architecture to accommodate multiple QoS requirements supporting several kinds of service models. The most successful or widely known among these mechanisms and models are the integrated services model (IntServ), the differentiated services model (DiffServ), traffic engineering, multi-protocol label switching (MPLS), and constraint-based routing.

1.2.1 Integrated Services Architecture

The integrated services architecture (IntServ) [BCS94] is characterized by resource reservation. It suggests that the setup of each communication flow needs to be signaled to all routers along the communication path which then, if still possible, reserve the desired bandwidth. IntServ is a very comprehensive proposal encompassing mechanisms such as the signaling RSVP protocol (resource reservation protocol) [BZB+97], the admission control routine, the classifier and the packet scheduler.

In addition to the traditional best-effort service, the IntServ model supports two service classes, controlled load (CL) service and guaranteed service (GS). These two classes are introduced specifically for interactive applications generating time-sensitive traffic. IntServ is based on the philosophy that guarantees cannot be achieved without explicit reservations. This model in consequence requires flow-specific state information in the routers and it is heavily dependent on the signaling resource reservation protocol RSVP that is demonstrated in Figure 1.1. While attempting to start a communication, the sender sends out a PATH message to the receiver, which includes the traffic characteristics. Every intermediate router along the communication path forwards the message to the next router. Upon receiving the PATH message, the receiver uses the traffic characteristics contained in the message and computes the capacity requirements for the desired QoS. It then responds with a resource reservation request (RESV) message, containing the capacity requirements. The RESV message is transmitted to the original sender along the reverse path of the PATH message. Every intermediate router along this path examines the RESV message and decides whether to accept or reject the reservation request depending on whether it can accommodate a new connection or not. If one of the intermediate routers rejects the reservation request, an error message is sent to the receiver and the signaling session is terminated. If all routers accept the request, the respective capacity resources are reserved and the flow state information is stored in the routers. Enhancements and extensions to the RSVP protocol are proposed in [GBH97] [LR98] [AGL+98].

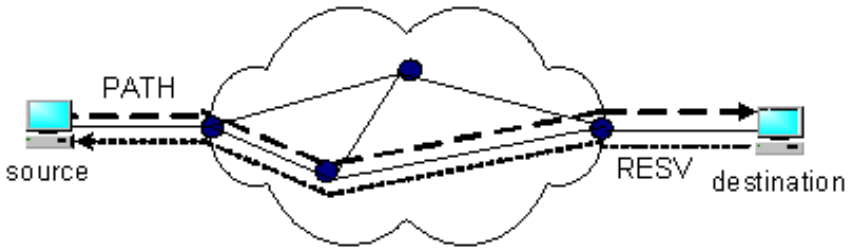


Figure 1.1: Resource reservation protocol.

The controlled load service [Wro97] aims to provide a service similar to that offered by best-effort service under lightly loaded networks. The CL service is qualitative in nature. It performs admission control and keeps the number of admitted flows under a certain limit in order to maintain the lightly loaded behavior of the network. The only guarantees of this service are that occasional bursts can be sent at a given peak rate and that the sustained transmission rate does not go below the given mean rate. The controlled load service does not provide any loss or delay guarantees.

The guaranteed service [SPG97], on the other hand, is designed for realtime communications requiring fixed delay bounds. It provides hard deterministic service guarantees. Those guarantees result in lossless transmission for conforming flows and firm end-to-end network delay, hence emulating a dedicated circuit for each of the flows. Since a lost packet corresponds to a packet with infinite delay, a bound on the delay limit implies a lossless transmission. GS merely controls the maximum delay; it does not attempt to minimize jitter or even the average delay.

The IntServ model embodies several limitations. The most important are the following.

- Per-flow state information needs to be stored in each associated router. Thus, memory requirements increase proportionally with the number of flows leading to scalability problems especially in the internet core.
- Routers must be able to run the RSVP protocol, carry out admission control mechanisms, multi-field classification, and packet scheduling.
- End-to-end deployment of the above mentioned mechanisms is mandatory for the GS model. Partial deployment is possible for the CL service where the RSVP protocol for example can be deployed at the bottleneck nodes of the network domain while the RSVP messages are tunneled over the rest of the domain.

1.2.2 Differentiated Services Architecture

The IETF working group has introduced the DiffServ model as a simpler means to provide scalable differentiated services in the IP network.

DiffServ [BBC98] is based on a simple model where all frames entering the network are analyzed and classified at the network edge. They are assigned to different behavior aggregates

(BA), which define the way in which frames are handled by nodes within the network core. As a result, DiffServ requires that analysis, classifications, marking, policing, and shaping operations be carried out at network boundary nodes only. The internal nodes have the mere task of examining a certain field in the IP header of the packet and determining accordingly the per-hop-behaviour (PHB) that defines the way in which they should forward this packet. This fact about DiffServ model is the key to its scalability. DiffServ has defined a 1-byte differentiated services (DS) field that replaces the type of service (TOS) byte in IPv4 and the equivalent traffic class (TC) byte in IPv6. The DS field consists of a 6-bit differentiated services codepoint (DSCP) and two bits that are currently unused. DiffServ is a relative-priority scheme. Three PHBs are standardized: expedited forwarding (EF), assured forwarding (AF), and the default PHB.

Expedited forwarding [JNP99] is often defined as a premium service since it constructs a virtual leased-line service type. EF PHB provides a low loss, low delay, low jitter, and assured bandwidth end-to-end service within DS-enabled domains. This premium service is achieved by guaranteeing that the departure rate of the EF aggregate of packets exceeds a certain configurable rate which is set by the system administrator. The EF traffic should receive at least the configured rate at the DS-enabled nodes irrespective of the intensity of the traffic. Several scheduling mechanisms can be employed to implement EF PHB. The simplest is strict preemptive priority queuing. Whenever EF traffic is received, all other transmissions are aborted and EF traffic is serviced immediately. In order to limit the negative impact this kind of traffic could have on other traffic, a rate policer or controller can be used for restricting the peak rate and burst size of the EF traffic.

Assured forwarding [HBW99] provides different classes of forwarding assurance for IP packets. Currently four classes are defined where each class is allocated a certain amount of resources at any DS-enabled network node. An AF class may receive more resources than allocated whenever excess resources are available. Within each AF class, IP packets are assigned different levels of drop precedence. In case of network congestion, a packet with higher drop precedence has a higher probability of being discarded than another packet of the same class with lower drop precedence. It is inferred that an AF implementation attempts to avoid long-term congestion by dropping packets while it allows short-term congestions, which result from bursts. Currently, three levels of drop precedence are defined.

Default PHB corresponds to the current best-effort forwarding treatment and it should be supported by all DS-compliant nodes. The network does not provide any guarantees to packets belonging to this aggregate, it delivers as many as possible and whenever possible. In order to ensure that other PHBs do not cause this kind of traffic to starve, some minimal resources (buffer and bandwidth) should be reserved. Any packet not belonging to any aggregate is marked with the default PHB codepoint.

In order to receive any of these PHBs, customers negotiate service level agreement (SLA) with their internet service providers (ISP). An SLA specifies the service class to be provided and the amount of traffic allowed for each class. It is defined either statically or dynamically. Static SLAs are discussed on a regular basis (e.g. monthly or yearly). Dynamic SLAs, on the other hand, are requested on demand by means of a signaling protocol such as RSVP. The SLAs determine the buffering, classification, policing, and shaping rules used at the network ingress. When a packet leaves one network domain and enters another, the new domain re-marks its DS field as determined by the SLA between the two domains.

It may happen that the IntServ model is deployed from end-to-end along a number of network elements that might be more complex entities than individual nodes. Complex entities can even be DiffServ networks for example. The IntServ model should then guarantee an end-to-end QoS across the given network, which can contain one or more DiffServ regions. This combination of the two proposed QoS models may facilitate the deployment of critical applications such as IP telephony where IntServ enables reserved resources along an end-to-end data path and DiffServ enables scalability across large networks.

In order to implement such a hybrid QoS model, the DiffServ regions should meet several requirements [BFY00].

- It should be able to support IntServ services within its domain. It should be possible that the boundary nodes of a DS-domain map the IntServ requests onto a DSCP that will invoke the appropriate PHB in the DiffServ network.
- It should provide admission control information to other network regions.
- It should be able to carry RSVP messages across the DS-enabled domain.

1.2.3 Traffic Engineering

Traffic engineering is the process of controlling the way in which traffic flows through the network so as to optimize resource utilization and performance of operational networks. It aims at avoiding network congestion caused by unbalanced network utilization. The major motivation behind the development of traffic engineering mechanisms is to facilitate reliable network operations such as enhancing network integrity, emphasizing network survivability, and increasing network robustness against service outages.

Network congestions are mainly caused by lack of resources or unbalanced distribution of traffic. Therefore, the network infrastructure has to be upgraded appropriately while at the same time the traffic load has to be evenly distributed among several paths in order to maintain load sharing. Unbalanced traffic distribution can occur due to traffic variation and the dynamic computation of routing paths using the current routing protocols such as RIP (routing information protocol), OSPF (open shortest path first) and IS-IS (intermediate system - intermediate system). These protocols select the shortest paths to forward packets. This way, it might happen that routers and links along these shortest paths become overloaded although those along a longer path are idle. The equal-cost multipath (ECMP) feature of OSPF and IS-IS is introduced to provide load sharing among several equal cost shortest paths.

The optimization objective of traffic engineering is the continual enhancement of network performance. This objective changes over time as new technologies emerge or as new requirements are imposed. The optimization aspects are ultimately concerned with capacity and traffic management. Capacity management includes planning, routing control, and resource management while traffic management includes traffic conditioning, shaping, queue management, and scheduling.

To optimize their resource utilization and network performance, internet service providers deploy traffic engineering mechanisms in their networks. Effective traffic engineering is achieved by introducing techniques such as MPLS and constraint-based routing.

1.2.3.1 Multiprotocol Label Switching

Multiprotocol Label Switching (MPLS) is a "base technology expected to improve the price and performance of network layer routing, improve the scalability of the network layer and provide greater flexibility in the delivery of routing services" as intended by the IETF working group. MPLS represents a forwarding scheme evolved from Cisco's tag switching. It has been referred to as *Layer 2.5* because it is inserted between the data link layer (layer 2) and the network layer (layer 3) [RVC01].

In conventional IP forwarding, decisions are based on the analysis of the packet's IP header and the results of the routing algorithm. This is basically done at each router. Using MPLS, a packet is examined at the network ingress where it is assigned a certain label. At subsequent hops, the packet's network layer header is not further analyzed. The label is used as an index into a table to directly determine the next hop and the new label to be used at the next hop. This leads to several important advantages of MPLS. Some of these advantages are listed below.

- Faster packet classification and forwarding: Routers and switches within a given network domain are capable of directly forwarding an incoming packet based on a fixed-length label. This label is used as a prefix in a lookup table that determines the next hop. No matter how complex the routing algorithm is, the router merely forwards a labeled packet. The complex process of classifying the packet and running the routing algorithm to determine its routing path is performed once at the network ingress. This feature of MPLS has become less significant as the performance of current routers is continuously improving.

- More flexible routing: MPLS adds more flexibility in determining the next hop of a packet. With conventional routing, the identity of a packet's ingress router is not carried with the packet and forwarding at the subsequent routers is based solely on the information available in the packet's network layer header. By means of MPLS, a packet entering a network at a particular router can be assigned a different label than the same packet entering the network at a different router. As a result, the routing paths of the packet differ when admitted at different routers.

- Explicit routing: This is one of the most compelling uses for MPLS and it is an important aspect for traffic engineering purposes. It gives network administrators the capability to identify explicit paths for given packets through their network domain based on any arbitrary criteria. This can lead to enhanced service quality in the network and provisioning of differentiated services [FWD+02]. Several parallel paths may exist from one end of a network domain to the other. If an IP telephony session is started for example between these two ends of the domain, the telephony packets can be assigned to the higher-speed, lower-delay path while other packets belonging to non-realtime applications can be assigned to a lower-speed, higher-delay path. Path selection can be performed with consideration of completely different criteria such as source address, destination address, per-flow characteristics, or even in response to RSVP messages.

1.2.3.2 Constraint-based Routing

Constraint-based routing computes routes subject to several constraints such as bandwidth, path length, network topology and other administrative policies. Constraint-based routing is an extension of QoS routing that selects the route that most closely meets the QoS requirements of

the given flow. Constraint-based routing may find that a long but lightly loaded path is better than a heavily loaded shortest path for certain traffic flows. Hence, network traffic becomes more evenly distributed.

In short, the objectives of constraint-based routing are

- selection of routes that can meet the required QoS criteria of flows,
- improvement of network utilization.

Constraint-based routing has also some drawbacks that are listed below.

- Increased computational overhead: The fine granularity of routing decisions leads to more flexibility and efficiency in terms of resource utilization and stability on one hand and increased computational and storage overhead on the other hand. Routing is performed based on various criteria whether destination based, source-destination based, class based or flow based.
- Increased routing table size: Finer granularity and more diverse route metrics increase the table size considerably. Conventional routing tables are two-dimensional arrays while the number of rows of constraint-based routing tables depend on the routing granularity and the number of columns on the route metrics. So, obviously, it may happen that the size of the routing table is far larger than the size of a normal routing table for the same network. This leads to increased storage overhead and slows down the routing table lookup process.
- Probable consumption of more resources for longer paths: Using a longer path for traffic routing consumes more resources than using the shortest path. This is acceptable when a longer path is lightly loaded at the time when the shortest path is heavily loaded. However, if the network load is heavy, it is inefficient to use the longer path. As a result, the widest path is favored at medium network loads while the shortest path is favored at heavy network loads.
- Probable routing instability: Constraint-based routing recomputes the routing table quite frequently, which may introduce instability. If routing is done on a destination basis and congestion occurred along the original route between two nodes, all traffic directed to that destination is shifted to an alternate route causing congestion to the alternate route. Traffic is then shifted again and so on. As a result, constraint-based routing should be deployed with caution especially in case of periodic recomputation of routing tables [AGKT98].

1.2.4 QoS Parameters

The term QoS lacks a definitive and unified definition. The ETSI (European Telecommunications Standards Institute) and the ITU (International Telecommunications Union) defined QoS as the quality perceived by the end user [ITU93]. The IETF network working group provided a more concrete definition where QoS is defined as the set of service requirements imposed on the network while transporting a traffic flow [CNRS98]. In this work, we adopt the latter definition and below we present the relevant metrics that constitute the set of service requirements. These metrics are used to assess the perceived QoS. Research efforts have been invested in formulating a relation among the QoS metrics to determine a final measure for the overall QoS and convert it into a subjective value [CR01] [Con02].

1.2.4.1 Delay

It is the elapsed time accumulating at each hop through the network from the sender to the receiver. Higher delay places more burden on the transport protocol to operate efficiently. For the transport control protocol (TCP) for example, higher delay implies that more data is being held in transit in the network and so performance stress is placed on the counters and timers of the associated protocol. The transmission rate of the sender is dynamically adjusted using the acknowledgements sent by the receiver. As these acknowledgement messages are delayed, the TCP protocol becomes more insensitive to network changes and the transmission rate of the sender is not adjusted in a timely manner. In case of the realtime protocol (RTP) which is used for interactive voice and video applications, higher delay causes more severe problems and possibly even interruptions as the whole system becomes unresponsive. Realtime applications are extremely sensitive to delay. They require deterministic playback of packets at the receiver to maintain the interactive communication between the users. Packets that exceed a given delay limit are considered lost; when arriving at the receiver side, these packets are discarded.

1.2.4.2 Jitter

Jitter is the variation in the end-to-end transit delay of the packets sent from the sender all the way to the receiver. It negatively affects network performance. Jitter is likely caused by network congestion, timing drift or route changes. High jitter affects the TCP protocol by making it operate inefficiently where it makes very conservative estimates of the round-trip time. High jitter has a more significant impact on realtime applications such as IP telephony and videoconferencing and it makes communication unacceptable. It causes the signal to be highly distorted. To cancel the jitter effect, a de-jitter buffer is placed at the receiver side to collect and store incoming packets that are then played out deterministically as desired. However, for high levels of jitter, a large de-jitter buffer is required and packets are forced to wait long. This contributes to the overall delay of the traffic making interactive communications very cumbersome to maintain.

1.2.4.3 Packet Loss

Packet loss occurs mainly due to buffer overflow. If buffers of a network node are full, all incoming packets incident at this node are discarded and lost. Packet loss affects both non-realtime and realtime applications and negatively impacts their performance. Data transmission is slightly affected by packet delay and jitter as mentioned earlier but it is extremely sensitive to packet loss. It requires a correct receipt of all its packets. Due to its valuable feature of reliable transmission, the TCP protocol is used for data transmission where it detects packet loss and issues a retransmission of all lost packets. However, high loss may cause a severe slow down of the overall transmission process. For realtime transmission, retransmission of packets is not possible and so high packet loss will dramatically decrease the perceived quality due to loss of information. It is worth noting that, in realtime transmission, packets with high delays exceeding the acceptable threshold are considered as lost and they add to the total number of lost packets.

1.2.4.4 Blocking Probability

While packet delay, jitter and loss are associated with packets only, there exist few metrics that evaluate the performance quality at the flow level like blocking probability. This probability is

the percentage of calls that arrive to the network and find no sufficient resources to get started. This metric is extremely relevant for realtime services and it has been a significant QoS metric for (virtual) circuit-switched networks.

1.2.4.5 Throughput

Throughput evaluates the performance quality at the flow level also. It is a main QoS measure for data traffic with non-realtime nature. Non-realtime applications are slightly affected by the time it takes the individual packets to be transmitted from source to destination. They are however extremely affected by the overall transfer time of the whole file until the last byte is received. Throughput is computed as the total size of the file divided by the transfer time. As a result, if high packet loss occurs then data throughput is dramatically reduced since the TCP window size is reduced by half.

1.3 Network Planning

All-IP converged networks should be planned with caution to accommodate new services with high and stable quality level. The ultimate objective of network planning is to lay the foundation for making profit out of network operation. This is achieved by granting a given network sufficiently high but *not too high* performance quality. Network planning examines the tradeoff between performance quality and resulting costs. Its task is to select the scenario with the optimal tradeoff, i.e. the one with the best quality for minimum cost. In summary, the objectives of network planning are the following.

- Economical network ownership: building and properly running a network is a very complex matter whose costs are difficult to evaluate. In order to estimate the network cost ownership, three categories are commonly differentiated: capital expenses (CAPEX), implementation expenses (IMPEX), and operational expenses (OPEX). CAPEX relates to the investment in network solutions, infrastructure and terminals. Underutilization of network resources means loss of investment. IMPEX relates to the costs of building the network such as civil works, installation, licenses, permits, etc.. Finally, OPEX includes expenses relating to costs of daily operations such as building rental, operations and maintenance, management and salaries, marketing, etc.

- Guaranteed quality of service: the ultimate goal is to provide a guaranteed QoS that achieves customer satisfaction. QoS as defined by ITU and ETSI organizations is a perceptual matter determined by the user making it very difficult to evaluate. The IETF working group assumes a rather simple decomposition of QoS into service requirements to make it more manageable and quantifiable. As mentioned earlier, service requirements at the flow level may include packet delay, delay jitter, packet loss, and throughput. Depending on the given service, different QoS measures are addressed. For realtime services, packet delay and jitter are the most relevant measures while for non-realtime services, packet loss and throughput are the most relevant ones.

With these objectives, a general network design problem can be formulated to determine the optimal values of variables such as topology, routing table and scheme, link capacities, bandwidth allocation and domain design. The network design problem can be stated as follows, where *Decision Variables* are the unknowns whose optimal values have to be found, *Objective*

Function is a mathematical function that represents the cost or the revenue for any proposed solution, and *Constraints* define the set of requirements for an acceptable solution.

Decision Variables: topology, routing, capacity assignment and domain design.

Objective Function:

$$Z = \min\left(f\left(\mathrm{CAPEX}, \mathrm{OPEX}, \mathrm{IMPEX}\right)\right). \tag{1.1}$$

Constraints:

- QoS criteria imposed by the supported services
- Robustness subject to traffic variations
- Robustness subject to node and link failures
- Constraints imposed by the applied routing scheme
- Others ...

The solution of this general problem is complex and not yet feasible due to the interdependencies among the different design variables. As a result, most research in this area is focused on solving subproblems. In the literature [Kle75], four basic design subproblems are defined: the flow assignment, the capacity assignment, the capacity and flow assignments, and the topology, capacity and flow assignments. Flow assignment determines the optimal routes over which information is transferred among the communicating nodes, capacity assignment determines the link capacities required for high quality transmission at minimum cost, and finally topology assignment determines node locations and link selection.

In addition to the general network design problem, network planning includes network management that aims at monitoring the network and upgrading it based on its performance. For example, routing tables are updated whenever a link or node failure occurs.

1.4 Thesis Contributions and Structure

Before highlighting the main contributions of this thesis, we briefly sketch our proposed network planning approach for realtime services in Figure 1.2. A given network with a fixed topology and link capacities might not be qualified for a diverse set of applications, each having its own QoS criteria. Starting new services in a network imposes quality demands and consequently requires enough link capacities. As a result, one objective is to efficiently map the QoS criteria of the different services to explicit capacity values at the network elements.

The starting step of this network planning process is first to examine the traffic characteristics generated by the applications at hand. At this point, an essential set of inputs is ready and it constitutes

- actual network topology together with available link capacities,
- complete set of services offered/to be offered,
- expected traffic load per service exchanged among network elements and represented by a traffic matrix,

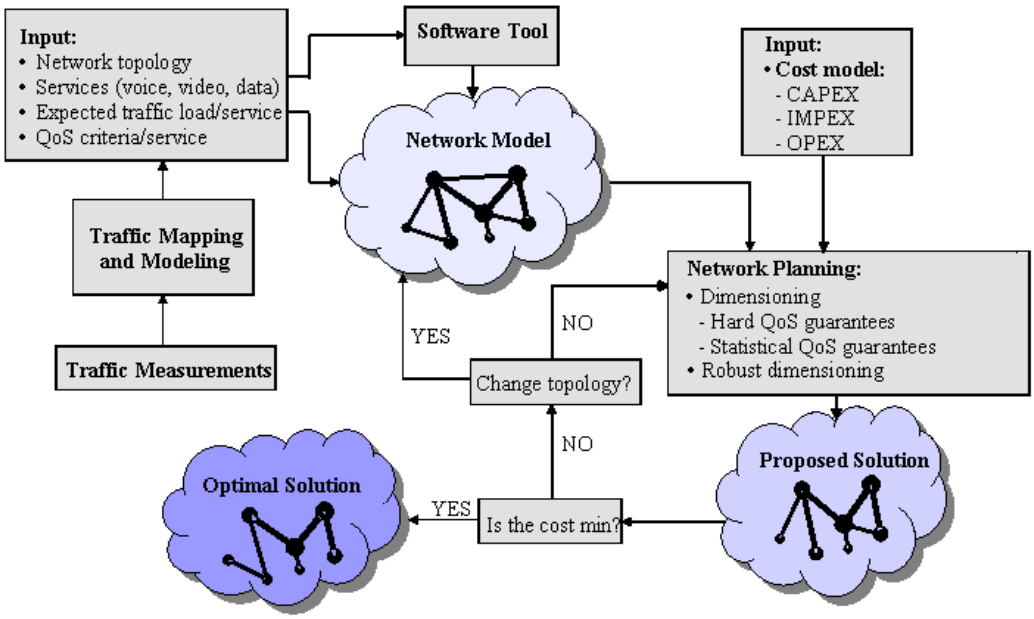


Figure 1.2: The flow process of the proposed network planning approach.

- QoS criteria per service,
- traffic specifications per service.

The network model can then be constructed using the previous set of inputs entered into a software tool. However, as our aim is to provide economical network operation with guaranteed QoS, we should inevitably account for the expenditures required for such a network. For this reason, we define a cost model that includes CAPEX expenditures and more specifically the cost of link capacities. In this dissertation, we aim for proper planning of existing data networks to offer realtime services. As a result, IMPEX and OPEX costs are not relevant as they are not affected by our planning intentions.

Now that the network and the cost models are ready, the core phase is started. Proper dimensioning methodologies are deployed to compute the required capacity shares for the offered services. Thereafter, the optimization process starts to find the appropriate network topology and the link capacities that minimize the cost. It might happen that changing the network topology provides a more economical solution in terms of link capacity costs. But if all expenditures are taken into account, changing the network topology might be rather expensive and not affordable. The option of changing the network topology is usually not desirable for network providers irrespective of public or private networks. Therefore, we leave out this option. It is worth mentioning that despite the objective of minimizing the total costs, the final network has to be robust enough against traffic deviation and imprecise traffic forecasting. Both problems are most likely to occur especially in IP networks which are generally loosely controlled. The iterative loop depicted in Figure 1.2 is an abstract representation of the optimization process performed during the planning phase.

1.4.1 Thesis Contributions

The main significance of this thesis lies in different areas of network planning, namely traffic characterization and control, network dimensioning and service capacity assignment, and network planning tool development.

1.4.1.1 Traffic Characterization and Control

Before any planning process takes place, it is crucial to determine the traffic characteristics and requirements. The current internet traffic with its mainly non-realtime nature differs significantly from realtime traffic which is the main focus of this dissertation. Even for the same application, the nature of traffic over different technologies may differ. The mean holding time of a PSTN/ISDN telephone call for example lasts between 90 and 120 seconds while an IP telephone call may last much longer in several cases due to the reduced cost. With regards to traffic characterization and control, this dissertation presents the following findings.

- Various VoIP applications are tested in different situations and the generated traffic is measured and characterized with respect to a known traffic model. While VoIP coders usually produce quite deterministic traffic behavior, it has been observed that traffic flows sent by software-based clients running under highly loaded operating systems show different properties. They inject flows which are very “out-of-profile”. In this dissertation, we propose various economical ways of accommodating these flows.

- A traffic policer on a flow basis is essential to adjust, control, and protect traffic admitted into the network.

1.4.1.2 Statistical Capacity Assignment for Realtime Services

To attain high quality performance, QoS guarantees should be provided for realtime services. Hard guaranteed quality levels, which assure deterministic delay bounds, incur extremely high costs. In this dissertation, we exploit the issue of statistical QoS provisioning and introduce two novel approaches of network dimensioning for interactive voice and video services.

Voice Service

- We propose a capacity allocation method that provides statistical performance guarantees to interactive voice service over IP networks. The proposed method provides slightly softened guarantees with some controlled tolerance to service degradation in order to notably reduce the amount of required capacity. This method is based on a novel concept: by providing statistical quality guarantees to those packets that experience the maximum waiting time among all packets of the active voice connections, all other packets are implicitly protected from excess delay and, thus, from service degradation.

- For the mathematical analysis of the method, we employ the basic analysis-synthesis approach where we decompose the problem gradually into a simple but not “simpler” subproblem, which can be mathematically handled. Thereafter, we build upon this simple model and generalize the mathematical solution to include the overall network model and account for the coexistence effect of other traffic types.

- A nonlinear optimization problem is formulated in order to obtain network-wide dimensioning results. The total link capacity shares allocated to the given service are minimized subject to performance constraints in terms of end-to-end delay requirements. We introduce a new extension to traditional capacity assignment problems by setting a statistical bound on the maximum end-to-end delay, which is experienced by all packets associated with the given voice service.

Video Service

- Video transmission over IP requires very high data rate for a good performance. Variable bit rate coded video is more popular than constant bit rate coded video due to its statistical multiplexing gain and stable quality level. This phenomenon forms a real challenge for network planning and especially for appropriate capacity evaluation. Video traffic shares common characteristics with that of voice but it still has its own inherent features. This fact lends the voice capacity allocation method as not suitable for video capacity allocation and thus a customized method is investigated. This method uses the aggregate flow of the given individual flows and assumes a mathematically lossless model (i.e. infinite buffer model). However, it accounts for unacceptably-delayed packets, which effectively add to the packet loss statistics. The assigned capacity share for an aggregate flow is set as a function of its traffic characteristics.

- A generic dimensioning model is derived that can be adapted to the different input parameters such as the quality level and the desired delay threshold.

Robust Network Dimensioning

IP network planning based on a given traffic demand becomes obsolete as soon as the actual traffic deviates slightly from the given values. It is highly probable that this be the case in IP networks with no deployed resource control mechanisms.

- In this context, we propose a novel way for provisioning a sufficiently robust network. This is done by accounting for traffic deviation inside the network while dimensioning this network.

- For robust network dimensioning, we introduce the concept of a capacity margin that is added to network links in order to account for certain traffic demand variability. Overall, the goal is a statistically robust network.

1.4.1.3 Network Planning Tool

In an integrated IP network, the supported traffic classes are mapped appropriately to the available network services provided by the actual technology of the network to grant the associated traffic its desired QoS. Each traffic class with a given load consumes part of the available link capacity. As a result, one can view the network link as a bundle of sub-links, each assigned a share of the total link capacity corresponding to the need of the associated class. According to the respective QoS measures, appropriate dimensioning strategies are to be used to adequately allocate network resources for each class of traffic. In this respect, our main contributions lie in the following issues.

- A software planning tool is developed with a generic layered architecture. Each layer is assigned a number of tasks independently from other layers. This tool can easily accommodate new traffic classes.
- It serves as a prototypical implementation for a software tool used for dimensioning converged IP networks with state-of-the-art algorithms that account for the major limitations in successfully deploying realtime services.

1.4.2 Thesis Structure

In Chapter 2, traffic characteristics of different realtime applications are determined through monitoring actual calls in different situations. The token bucket parameters have been utilized to quantitatively describe traffic characteristics and to calculate the corresponding bandwidth requirements. Most VoIP clients generate quite deterministic and pretty smooth traffic except for software-based clients, which run under non-preemptive operating systems without special countermeasures. Software-based clients generate very bursty traffic in case of highly loaded situations. In this chapter, we propose ways of greatly minimizing the negative effect of such cases with rather economical solutions. Based on a general network architecture, we analyze the effects of traffic characteristics mainly from the perspective of network dimensioning.

In Chapter 3, we propose a new dimensioning strategy for interactive voice communications required to provide "almost" guaranteed QoS with rather low costs. We investigate this method in an analysis-synthesis approach constituted of several levels. The analytical solution of the basic model is extended from one level to another to arrive at the general network model (the converged IP network). At this level, the problem of capacity assignment is formulated as a nonlinear optimization problem. This is solved by means of link decomposition and known mathematical optimization techniques. Finally, a generic network planning tool is developed to solve the presented capacity assignment problem in a timely manner.

In Chapter 4, we investigate a new dimensioning model for the interactive video service using real video files. The model allows for statistical QoS guarantees by defining an outage probability which determines the percentage of video frames that are allowed to exceed the desired delay threshold without affecting the needed quality level. We investigate the case in which the requested delay threshold affects the capacity share needed to abide by the given threshold. Finally, we examine the impact of changing the coding quality level of the given videos on the results obtained by the capacity assignment method.

In Chapter 5, we handle the issue of network dimensioning for stream traffic under uncertain demands and absence of complex resource control mechanisms. Based on a practical model, we statistically study the distribution of traffic demand inside the uncontrolled network region. We introduce the concept of a capacity margin which is needed in addition to the planned capacity value to account for a certain level of traffic demand variability and to achieve high degrees of robustness.

In Chapter 6, a brief summary of this work is presented and the main results are discussed. Thereafter, open research issues with interesting objectives are highlighted.

Parts of this thesis have been published in [SRGT03], [SRT03], [Sha04], [SD04], [SKD04], [SRB05], and [SD05].

2

Traffic Characterization and Capacity Requirements for Individual Sources

The planning of broadband multiservice networks depends on traffic characteristics and performance requirements of the various services to be supported across the network. In a packetized network integrating telecommunications services such as telephony and videoconferencing characterized by interactive realtime communications and stringent performance requirements, it is highly important to characterize, monitor, and control the incoming traffic. This is needed to provide the sufficient resources for the given traffic and to restrain any ill-behaving traffic from damaging the quality of other traffic belonging to the same class of service or even to different classes of service.

An essential motivation for migrating telecommunications services to packetized networks is that such networks efficiently share their resources between variable bit rate traffic streams. Moreover, sources in packetized networks can be made to “transmit necessary information when necessary”, i.e. sources can be made intelligent enough to send only integral and non-redundant information allowing for better efficiency. In video transmission for example, major gain in capacity is achieved if only the difference between one scene and the other is coded and transmitted rather than coding and transmitting the whole scene whether a slight change has only occurred or a major one. This can also be applied in voice transmission if silence periods during a telephone conversation are suppressed. In consequence to the “transmit necessary information when necessary” strategy, the generated traffic is made rather difficult to predict thoroughly especially that the nature of the communication is realtime. However with the use of rate control algorithms in the source coders and with some prior knowledge about the traffic activity, the generated traffic can be statistically modeled.

In order to make sure that traffic entering the network abides by certain characteristics, a traffic policing unit is used at the network ingress to control incoming traffic using rule-based parameters. It is quite infeasible to capture all traffic characteristics by few parameters. However, according to the nature of traffic and its intrinsic properties and requirements, the most relevant traffic model can be formulated. It is often required to have a unified traffic description in order to allow the integration of different services in one network. Doing so, the network has a common "rule" and can thus handle different traffic flows appropriately according to their class of service. This particular rule is commonly the leaky bucket that is known variously as the virtual scheduling algorithm, the generic cell rate algorithm, and the token bucket filter. The generic cell rate algorithm was first defined by the ATM Forum [For96] by generalizing the peak cell rate definition given in [ITU96e]. The token bucket model is closely related to the former algorithm though applied to the more general context of variable length packets, which makes it more suitable for IP networks. The token bucket model has been adopted by the ITU-T in H.323 standard for describing traffic characteristics for multimedia communications over packet-based networks.

During the last years, lots of researchers have investigated the feasibility of audio and video transmission over IP networks using real or simulated scenario measurements by looking at different aspects of the deployment. Various studies [MTK03] [MT02] [MCPA01] have focused on examining the delay and loss behaviors over the internet and their impact on the perceived quality. [JS99] addresses the problem of modeling network delay and loss through internet measurements. In [RASHB03], one-year measurements are conducted over local and domestic internet sites to study the loss behavior, delay, and jitter of audio sources and to understand the trade-offs among various encoders. Other studies have addressed the issue of traffic specifications and policing of audio/video sources over the internet. In [BVJP02], the authors attempt to evaluate and compare traffic descriptors for aggregate traffic having constant bit rate sources and sources using the silence suppression. Others examine the differences in traffic among varying multimedia applications [CK02]. The distinction of this work from the previous studies lies in the following factors: (i) characterizing individual sources under different conditions for per-flow capacity evaluation where software-based clients are especially investigated (ii) dimensioning policing units so as to achieve guaranteed QoS with deterministic delay threshold and zero packet loss (iii) trade-off investigations among traffic specifications of various VoIP or video over IP clients to provide the most economical capacity requirements with desirable performance, where packet loss is introduced when necessary.

In this chapter, we attempt to characterize traffic traces generated by individual IP telephony and videoconferencing sources based on the token bucket (TB) model. The TB model is then used to dimension traffic policing units, which force the generated traffic to be transmitted at a more predictable rate, facilitating the planning process. We particularly consider different types of IP telephony or voice over IP (VoIP) user applications that need to be provided by a multiservice IP network and examine their traffic properties through real-scenario measurements. Based on the measured TB model parameters and the desired performance criteria, the capacity requirement of each regulated source is then evaluated. Moreover, we discuss the effects of tuning the traffic characteristics, mainly from a perspective of network dimensioning where possibilities are discussed of how the required capacity values can be reduced. As for videoconferencing applications, elaborate measurements are performed in a separate study [Gla03] [EG04]. At the end of this chapter, we briefly present the relevant results of [Gla03] [EG04] and use them for our analysis of video traffic characteristics.

This chapter is organized as follows. In Section 2.1, traffic is observed in different detail levels –call level, burst level, and packet level. For the purpose of estimating the capacity needs for individual flows, we are mostly interested in the packet level. Section 2.2 gives an overview about the characterization model used in addition to the dimensioning methodology applied for evaluating the per-flow capacity requirements. Section 2.3 presents the measurement results of different VoIP applications under various situations. It then covers trade-off considerations between service quality and capacity needs. Section 2.4 presents analogous analysis for video over IP sources and demonstrates the dimensioning results obtained with different setting parameters. Finally Section 2.5 concludes the chapter and brings the light on further problems that are handled in the following chapters.

2.1 Multi-Level Nature of Traffic

Traffic was first observed by Hui in [Hui88] with various detail levels: the packet level, the burst level, and the call level. This is possible due to the different nature of traffic at the three different levels, along with the decoupling of traffic characteristics at each of these levels. The three-level structure is illustrated in Figure 2.1. It shows that each call is composed of bursts, and bursts are subdivided into packets. A call is interpreted simply as a connection between two parties and it is initiated by a connection request signal. A burst can be interpreted as a burst of speech in voice communications and as a one-screen burst in case of video communications. We note that the flow of information during a call may not be continuous and so a more accurate term is a virtual call.

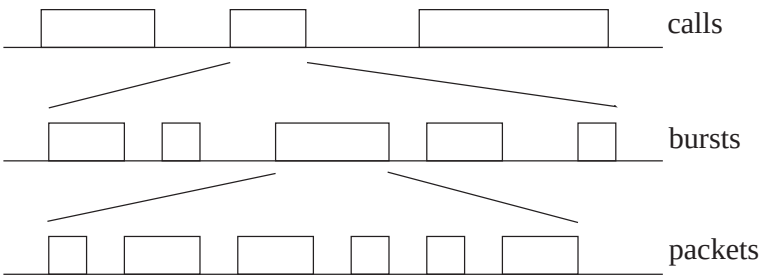


Figure 2.1: Three-level structure of traffic.

In general, any traffic stream may be observed in an arbitrary number of detail levels that can be determined depending on the traffic nature and the analysis purpose. It seems however that for most purposes, a three-level structure is often detailed enough. The very detailed observation can be the electric signal coupled with each bit of information and this can be viewed by means of an oscilloscope.

The attribute that differentiates and couples the different levels is basically the *time scale*. The time scale can be substantially different from one level to the other. At one level, the finer granularity of a lower level is ignored and the time scale is governed by the activity period of one entity in this level that is determined by the activity period of a sequence of entities at the lower level.

In this chapter, we aim for evaluating the capacity requirements for individual VoIP and video-conferencing sources so as to satisfy performance criteria mainly in terms of packet delay. For

highly interactive communications, the packet transmission delay from source to destination should be constrained within a certain threshold. As a result, we need to find the minimum capacity requirement of each flow such that all its packets are transmitted within the desired threshold at all times providing hard QoS guarantees. This makes us focus on traffic properties of VoIP and videoconferencing sources at the packet level and this is done here by means of the token bucket model.

2.2 Applied Methodology for Per-Flow Traffic Characterization and Dimensioning

In this work, we assume a network scenario as depicted in Figure 2.2. Different types of realtime applications generate traffic flows that are passed through policing units labeled as *TB* to make them fit to a pre-determined traffic profile. Thereafter, traffic flows are aggregated and sent over the IP network to their respective destinations. In case of a gateway, which virtually represents several traffic sources, a corresponding number of policers have to be installed, each controlling the characteristics of one individual flow.

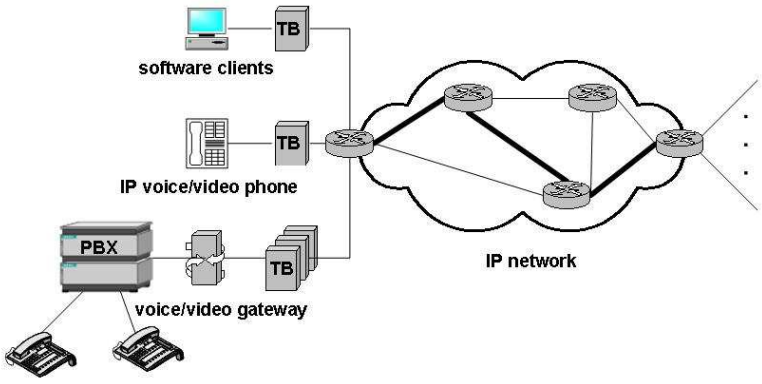


Figure 2.2: Network scenario.

2.2.1 Token Bucket Principle

The token bucket model can serve as a i) policing unit which regulates the source's output, ii) traffic shaper inside the network to smooth traffic between network nodes and iii) means of characterizing traffic in terms of a set of parameters which can be used to compute the required service rate. The service rate refers to the share of capacity needed by one traffic flow so as to meet its performance requirements.

In our network scenario in Figure 2.2, the TB entity depicted is a policing unit and it stands for "token bucket". Being between a multimedia host and the network ingress node, it ensures that the traffic entering the network conforms to a certain pre-determined profile. In case the traffic violates this profile, the TB policer drops incoming packets in a way to make the outgoing traffic fit the given profile. The basic token bucket policer is characterized by two parameters: the token fill rate and the bucket depth, and its operation is described as follows. The bucket

is continuously being filled with tokens at a constant rate (one token every fixed time interval) and it can hold up to a limited number of tokens. Every time a packet passes through the TB, one token is removed from the bucket. When the bucket runs empty, the arriving packets are dropped until new tokens are available in the bucket. As a result, the average rate of the outgoing flow is bounded by the token fill rate.

The implementation of the basic TB model can simply be a variable that counts tokens. As tokens appear at constant time intervals, the counter is incremented by one at each interval and decremented by one whenever a packet is transmitted. If the counter hits zero, no packets are allowed and are thus lost. This form of the token bucket is the token-count variant and it is mostly used in ATM (asynchronous transfer mode) networks where all packets (cells) have a common size. In a byte-count variant of the TB model, the counter is incremented by a fixed number of bytes every fixed interval of time and decremented by the packet size each time a packet is transmitted. The byte-count variant is used in IP networks and it is the one adopted in this work.

The operation of the TB policer is depicted in Figure 2.3 where r is the token fill rate and b the bucket depth. In the token-count variant, r and b parameters are given in tokens per second and tokens respectively; whereas, in the byte-count variant, r and b parameters are given in bits per second and bytes respectively. A traffic source conforms to token bucket (r, b) if tokens are always available in the bucket whenever a packet is generated. The TB allows sources at idle intervals to save up permission to send larger bursts later (b tokens can be saved up in a bucket as a maximum limit). As a consequence, the largest amount of traffic, which can pass the TB in the form of a burst in an arbitrary time interval $\triangle t$, is equal to $b + r\triangle t$. The burst can be conceptually sent at an infinite rate that is in practice limited to the outgoing link capacity.

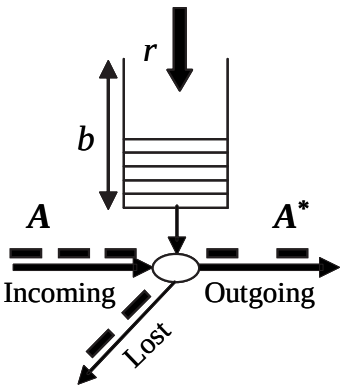


Figure 2.3: Token bucket operation.

A more complex variant of the TB policer has an additional capability of limiting the peak rate p of a source as desired rather than allowing it to take the value of the outgoing link capacity no matter how high it is. This variant is modeled by the set of parameters $\mathbb{S} = (r, b, p, m, M)$ with r being the token fill rate, b the bucket depth, p the peak rate, m the minimum policed data size, and M the maximum policed data size. As a result, the complex TB variant allows for a burst at a peak rate p as long as there are enough tokens in the bucket. In the sequel, TB model refers to the byte-count complex variant, which offers more control on traffic.

Denoting the characteristics of the incoming and the outgoing traffic of a TB policer with A and A^* respectively, we can formulate a relationship $\mathcal{T}$ where

$$A^* = \mathcal{T}_{\mathbb{S}}(A), \tag{2.1}$$

where $\mathbb{S}$ is the set of token bucket parameters.

2.2.2 Per-Flow Capacity Evaluation for Deterministic QoS Guarantees

For guaranteeing a desirable level of communication quality, an intelligent decision about the needed level of resources should be made. The amount of resources to be reserved can be evaluated as a function of source characteristics and network characteristics. By source characteristics, we refer to flow specifications –provided in terms of token bucket parameters– and to flow requirements –provided in terms of end-to-end delay, packet loss, and jitter. By network characteristics, we refer to factors such as the number of hops along the path followed by the given flow, the scheduling scheme employed at each hop, and the end-to-end latency occurring in the network.

In terms of (2.1), this means that for a traffic source, a set of TB parameters $\mathbb{S}$ has to be derived such that $A^* = \mathcal{T}_{\mathbb{S}}(A) = A$, i.e. all packets emitted by a multimedia application pass the token bucket without loss or delay. Based on the TB description $\mathbb{S}$ of traffic $A^* = A$, a capacity demand R is calculated which is denoted as the service rate. R represents the capacity requirement for one individual flow traveling from source to destination. To compute R, the envelope process concept is adopted [Cha94][Cru91] and a virtual circuit is assumed to be established from the source to the destination by means of the resource reservation protocol (RSVP) for example [SPG97]. RSVP is designed originally for IP networks that support the IETF's integrated services model.

Being monitored by a policer, traffic $A[s,t]$ that arrives at the network ingress in any given time interval $[s,t]$, for any $t \geq s \geq 0$ is said to be A-smooth and satisfies the inequality

$$A[s,t] \leq A(t-s) \qquad t \geq s \geq 0, \tag{2.2}$$

where $A(t-s)$ is called the envelope process. Inequality (2.2) defines the smoothness criterion and it is termed the "burstiness constraint". If the flow is TB-constrained, then by definition it has an envelope $A(t-s)$ given by

$$A(t-s) = \min\{M + p \cdot (t-s), b + r \cdot (t-s)\} \qquad t \geq s \geq 0. \tag{2.3}$$

Given a non-decreasing non-negative function $S(\cdot)$, a network element i is said to guarantee a service curve S if for any $t \geq 0$, there exists an $s \leq t$ such that there is no backlog of the flow at time s, and the service received by the flow in the interval $[s,t]$ is not less than $S(t-s)$ [Cru95]. If a rate R is reserved for a given flow at network element i, then the service

curve can be obtained from parameters ζ_i and η_i that represent the quality of service provided by this network element to all flows traversing it. Typically, these parameters capture the deviation of the network element from the hypothetical fluid server operating at rate R. The ζ_i term accounts for the packetization effect of bits, where the last bit of the packet has to wait till all the preceding bits of the packet are transmitted. The η_i term, however, accounts for the multitraffic effect, meaning that it measures the time the packet needs, once ready for transmission, to wait for the physical layer to be free. As a result, the service curve at network element i is given by

$$\begin{aligned} S_i(t-s) &= \left[\left((t-s)-\frac{\zeta_i}{R}-\eta_i\right)R\right]^+ \\ &= \left[\left((t-s)-\eta_i\right)R-\zeta_i\right]^+, \end{aligned} \tag{2.4}$$

where $[x]^+ \equiv \max\{x, 0\}$.

At this point, the service curve of a tandem of network elements $1, 2, \ldots, i$ along the path of the flow can be calculated as done in [Cru95] to be given by

$$\begin{aligned} \bar{S}_i(t-s) &= \min\left(\sum_{j=1}^{i} S_j(t_j-s_j) : (t_j-s_j)\geq 0,\ \sum_{j=1}^{i}(t_j-s_j)=(t-s)\right) \\ &= \left[\left((t-s)-\sum_{j=1}^{i}\eta_j\right)R-\sum_{j=1}^{i}\zeta_j\right]^+. \end{aligned} \tag{2.5}$$

Knowing the traffic envelope of a flow and the service curve $\bar{S}_i$, the upper bound $\widehat{\mathcal{D}}_{1\to i}$ on the delay incurred by the packets belonging to this flow from the time they enter network element 1 until they leave network element i can be given by the maximum horizontal distance between the traffic envelope and the service curve. Referring to Figure 2.4 [GGPR96] that plots both the traffic envelope and the service curve, $\widehat{\mathcal{D}}_{1\to i}$ can be readily computed as the magnitude of segment EF that evaluates to

$$\widehat{\mathcal{D}}_{1\to i} = \begin{cases} \dfrac{(b-M)}{R}\dfrac{(p-R)}{(p-r)}+\dfrac{M}{R}+\displaystyle\sum_{j=1}^{i}\left(\frac{\zeta_j}{R}+\eta_j\right) & \text{if } p>R\geq r, \\ \dfrac{M}{R}+\displaystyle\sum_{j=1}^{i}\left(\frac{\zeta_j}{R}+\eta_j\right) & \text{if } r\leq p\leq R, \end{cases} \tag{2.6}$$

where (r, b, p, m, M) are the TB parameters, R is the service rate, ζ_j and η_j are the individual error parameters of network element j. According to [PG93] [PG94], it follows that ζ_j and η_j parameters of any network element j using weighted fair queuing (WFQ) are given by

$$\zeta_j = M, \tag{2.7}$$

$$\eta_j = \frac{M_{\text{MTU}}}{C}+T_{\text{prop}}, \tag{2.8}$$

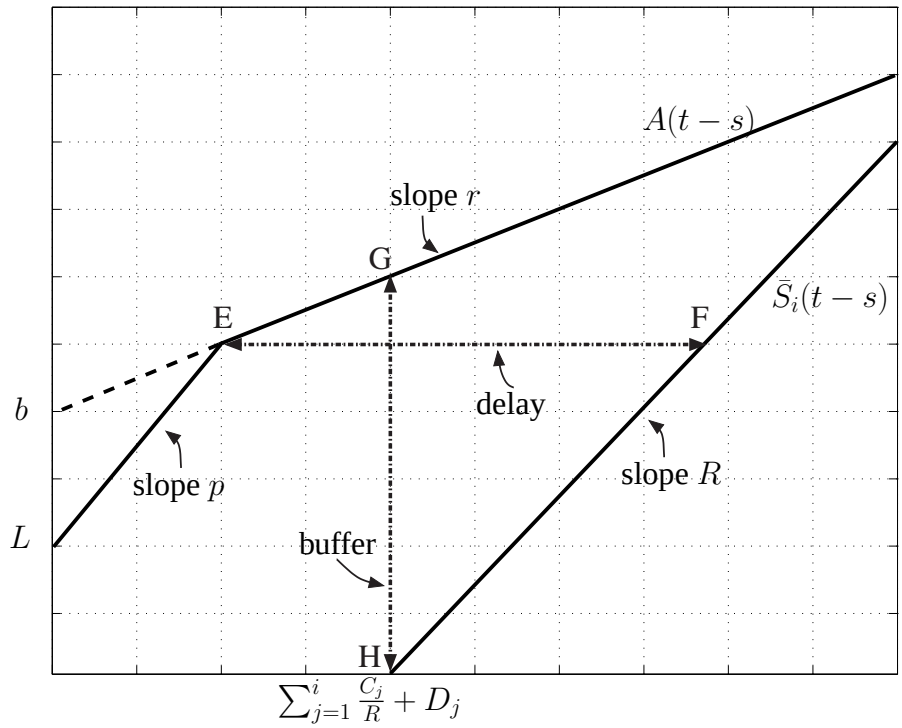


Figure 2.4: Delay and buffer calculations. x-axis is time in ms and y-axis is amount of data in bytes.

where M_{MTU} is the link maximum transfer unit (MTU), C the link capacity, and T_{prop} the propagation delay. WFQ is a scheduling scheme that incorporates several first-in-first-out (FIFO) queues where each queue is assigned a certain weight and the link capacity is shared among the busy queues in direct proportion to their assigned weights. The formulae in (2.6) can be solved for R in terms of the number of network elements traversed along the whole flow path and the desired end-to-end network delay bound $\widehat{\mathcal{D}}_{\mathrm{net}}$ as

$$R = \begin{cases} \dfrac{p \cdot (b-M) + (p-r)\left(M + \sum_j \zeta_j\right)}{b - M + (p-r)\left(\widehat{\mathcal{D}}_{\mathrm{net}} - \sum_j \eta_j\right)} & \text{if } p > R \geq r, \\ \dfrac{M + \sum_j \zeta_j}{\widehat{\mathcal{D}}_{\mathrm{net}} - \sum_j \eta_j} & \text{if } r \leq p \leq R. \end{cases} \tag{2.9}$$

In a similar manner, the buffer requirements B_i at network element i for a zero packet loss is given by the maximum vertical distance between the traffic envelope of the flow and the service curve (i.e. segment GH in the figure). Solving for B_i, we get

$$B_i = M + \frac{(p-X)}{(p-r)}(b-M) + \sum_{j=1}^{i}\left[\frac{\zeta_j}{R} + \eta_j\right] X, \tag{2.10}$$

where,

$$X = \begin{cases} r & \text{if } \dfrac{(b-M)}{(p-r)} \leq \displaystyle\sum_{j=1}^{i} \left(\frac{\zeta_j}{R} + \eta_j \right), \\ R & \text{if } \dfrac{(b-M)}{(p-r)} > \displaystyle\sum_{j=1}^{i} \left(\frac{\zeta_j}{R} + \eta_j \right) \text{ and } p > R, \\ p & \text{otherwise.} \end{cases}$$

Later in this chapter, (2.9) is used as the basis for computing the individual capacity requirements for given voice and video flows.

2.2.3 Approach for Network Dimensioning

At this point, the capacity demand R for each individual flow can be determined based on the TB description $\mathbb{S}$ of traffic $A^* = A$. Summing up the individual capacity requirements of all active flows gives the total capacity C_{class}, which has to be allocated to the given class in order to carry all flows with the appropriate QoS. From the perspective of call admission control (CAC), this means that if a certain capacity share is assigned to one traffic type, the maximum number of active flows belonging to this traffic type can be determined (only if individual flows are assigned equal service rates). Figure 2.5 depicts the described approach where A_i, A_i^*, and S_i denote the A, A^* and S of traffic flow i. R_i denotes the capacity requirement for individual flow i and C_{class} denotes the total link capacity share allocated for a given traffic class.

We note that no statistical multiplexing is considered in this chapter due to the fact that we are interested in evaluating the capacity requirements for deterministic delay guarantees. In Chapter 3 and Chapter 4, we propose and investigate new capacity assignment methods that consider the statistical multiplexing effect of the active traffic flows.

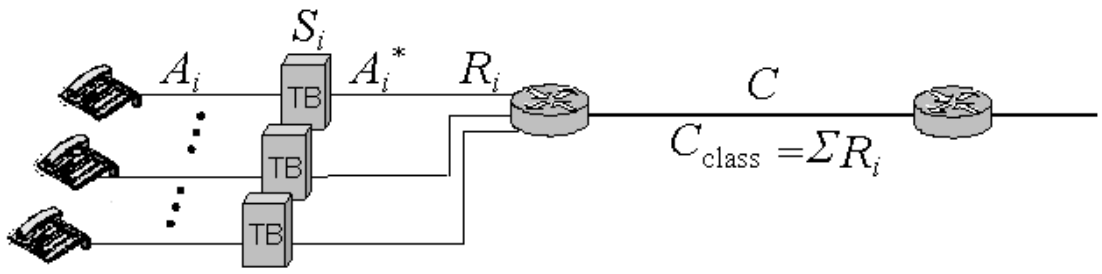


Figure 2.5: Approach for network dimensioning.

2.3 Interactive Voice Services

In this section, the token bucket parameters for different VoIP clients are determined using real-scenario measurements. Based on these parameters, the service rate R_{voice} of a single traffic source is computed. To do so, we assume that we have a sequence of three routers all using WFQ scheduling, link speeds of 100 Mbps, and an MTU of 1500 bytes. The network delay bound is set to $\widehat{\mathcal{D}}_{\text{net}}$ = 50 ms, and the propagation delay is ignored. We note that $\widehat{\mathcal{D}}_{\text{net}}$ does not account for the coder delay, which is dependent on the coder type and the number of frames packed in one RTP (real time protocol) packet.

2.3.1 Description of VoIP Clients

A VoIP client is defined as any application or equipment that is able to generate and receive IP-packetized voice traffic by means of certain coders. Based on this definition, VoIP clients encompass IP phones, soft clients, and VoIP gateways. IP phones are stand-alone terminals that produce voice packets which are eligible for transmission over an IP network. Soft clients emulate a similar function to IP phones, but they are software-based clients that could run on personal computers (PCs). One advantage of soft clients is providing an integrated platform for voice and data communications. VoIP gateways interconnect IP networks with circuit-switched systems such as traditional telephony networks.

Every VoIP client incorporates a voice coder, which is capable of transmitting and receiving voice traffic. A voice coder, also known as a codec, consists of an encoder and a decoder. The encoder operates on collections of speech samples and compresses them into frames. The decoder, on the other hand, receives the encoded frames and recreates the original speech samples. An encoder is said to compress 10 ms frames if it compresses, for example, blocks of 80 16-bit voice samples sampled at Nyquist rate and quantized using 16-bit levels knowing that the analog voice bandwidth is limited to 4 KHz, i.e.

$$\frac{80 \times 16 \text{ bits}}{8000 \, \frac{\text{sample}}{\text{sec}} \times 16 \, \frac{\text{bits}}{\text{sample}}} = 10 \text{ ms}$$

2.3.1.1 Voice Coders

The ITU-T has standardized a number of voice coders for encoding voice frequency signals. The traditional voice coder that is widely used on digital telecommunications networks is called G.711. However, in the last decade, new voice encoding algorithms have been developed using code-excited linear prediction (CELP) techniques, leading to drastic rate reductions at the expense of additional encoding delay. The transmission rate can be further reduced if silence periods are cut-off and not transmitted through the network. Voice conversations contain normally 35–50% of silence periods. Some coders have implemented a silence suppression scheme which consists of voice activity detector (VAD) and comfort noise generator (CNG). VAD is implemented at the encoder side to distinguish between speech samples and silence samples. Therefore, instead of sending VoIP packets of silence, VoIP gateways can interweave data traffic with VoIP conversations, and consequently the bandwidth could be effectively utilized with the expense of increasing traffic variability. The CNG, on the other hand, is implemented at the decoder side to reconstruct background noise so as to prevent mistaking silence for a

disconnected call. A brief description of the commonly used voice coders is presented below [ITU99b][KKS01].

G.711 –specified in [ITU72]– is widely used on digital telecommunications networks. G.711 has two recommended encoding laws and these are commonly referred to as the A-law and the μ-law. The former is employed in Europe while the latter is employed in North America and Japan. G.711 μ-law and A-law compress 14-bit and 13-bit uniform pulse coded modulation (PCM) samples respectively into 8-bit logarithmic samples producing a bit rate of 64 kbps.

G.723.1 –specified in [ITU96d]– was primarily developed for PSTN videophones. G.723.1 is optimized to represent speech with a high quality at very low bit rates with limited complexity, utilizing multi-pulse maximum likelihood quantization (MP-MLQ) at 6.3 kbps and algebraic code-excited linear prediction (ACELP) at 5.3 kbps. MP-MLQ compresses 30 ms blocks of 240 16-bit samples into 24-byte frames, while ACELP compresses 30 ms blocks into 20-byte frames. Each G.723.1 encoder and decoder should support both rates, 6.3 and 5.3 kbps. It is possible to switch between the two rates at any 30 ms frame boundary. [ITU96a] specifies a silence suppression scheme consisting of VAD and CNG.

G.726 –specified in [ITU90]– utilizes adaptive differential pulse code modulation (ADPCM) to convert a 64 kbps A-law or μ-law PCM channel (G.711) to a 16, 24, 32, or 40 kbps channel by compressing the 8-bit samples of G.711 into 2-bit, 3-bit, 4-bit, or 5-bit samples.

G.728 –specified in [ITU92]– encodes speech signals using low-delay code excited linear prediction (LD-CELP) that compresses 0.625 ms blocks of five 16-bit samples into 10-bit frames at the rate of 16 kbps. LD-CELP also has 12.8 and 9.6 kbps bit rate extensions [ITU99a].

G.729 –specified in [ITU96c, ITU96b]– utilizes conjugate-structure algebraic-code-excited linear-prediction (CS-ACELP) for coding speech signals at 8 kbps. CS-ACELP compresses 10 ms blocks of 80 16-bit linear PCM samples into 10-byte samples. CS-ACELP also has 6.4 and 11.8 kbps bit rate extensions [ITU98a, ITU98b]. G.729 has a reduced complexity version, G.729A, which is interoperable with the full version. G.729A has been developed for –but not limited to– simultaneous voice and data applications. The VAD and CNG silence suppression algorithms are specified in [ITU97].

There are several attributes that compare the performance of coders among each other. The basic attributes are presented below.

- Coder type: It is the algorithm used for encoding and decoding input speech signals.
- Bit rate: Reduction in bit rate results in economical use of link capacity at the expense of reduced quality. It is important to have low bit rates as long as a certain quality level is not compromised.
- Complexity: The coder complexity is measured by the number of millions of instructions per second needed to execute the encoder and the decoder algorithms. More complex coders consume more processing power and memory usage, thus, inducing more cost.
- Frame size: Voice coders operate on groups of speech samples known as frames to produce a compressed frame. The frame size is measured in time unit and it represents the time needed to collect speech samples to generate one compressed frame. The compressed frame is not generated until all samples in the group are fed into the encoder; thus, there is a delay equivalent to one frame size before processing starts.

- Look-ahead: Many voice coders do not start processing directly after having a complete frame; they wait to collect some samples from the succeeding frame to use for improving compression efficiency. This additional time is called look-ahead.
- Algorithmic delay: The time needed for the encoder to start processing a frame is known as the algorithmic delay which is the sum of the frame size and the look-ahead.
- Quality: The coder quality is generally measured using a comparative subjective rating called the mean opinion score (MOS). This rating evaluates the quality of a given coder as compared to an established reference coder and assigns an MOS score ranging from 1 to 5, where 1 is bad and 5 is excellent.

Table 2.1 summarizes the basic attributes of each of the presented voice coders [ITU99b]. The bit rate values in Table 2.1 refer to the coder bit rates if operating at a constant rate (i.e. without activating the VAD). The presented rates are different from the resulting IP layer bit rates that account for the additional headers of RTP, UDP (user datagram protocol), and IP protocols. The coder normally generates a voice frame every fixed time period. These frames are collected by an RTP packetizer that constructs the RTP packet and hands it to the IP network stack. The number of voice frames collected in one RTP packet is set at configuration time and can be changed.

Table 2.1: Comparison of voice coders.

Coder type	**G.711**	**G.723.1**	**G.726**	**G.728**	**G.729**	**G.729A**
Coding algorithm	PCM	MPC-MLQ / ACELP	ADPCM	LD-CELP	CS-ACELP	CS-ACELP
Bit rate (kbps)	64	6.3 / 5.3	16 / 24 / 32 / 40	16	8	8
Complexity (MIPS)	$\ll 1$	≤ 18	≈ 1	≈ 30	≤ 20	≤ 11
Frame size (ms)	0.125	30	0.125	0.625	10	10
Look ahead (ms)	0	7.5	0	0	5	5
Algorithmic delay (ms)	0.125	37.5	0.125	0.625	15	15
Quality (MOS)	4.0	3.9 / 3.6	3.6 to 3.9	3.6	3.9	3.7

2.3.2 Measured Traffic Characteristics of VoIP Clients

Modeling voice traffic in real network scenarios requires capturing the RTP messages exchanged between two communicating parties and analyzing them. The network analyzer Ethereal was used for this purpose [Eth01]. It is a passive measurement tool that listens to the network without injecting additional traffic. In order to determine the source characteristics without distortion, only two (in some cases three) hosts were connected to the measurement network. The hosts operate under Windows NT, however, Windows 2000 is also tested and similar results are achieved. Furthermore, background traffic, which could affect the voice traffic, was avoided.

In order to determine the capacity requirement for an individual voice flow regardless of the communication activity, the VAD option was deactivated in all clients causing the maximum traffic load on the network. It is also observed that with VAD activated, the communication quality is deteriorated due to the artificial background noise generation and the additional processing delay.

2.3.2.1 IP Phones and VoIP Gateways

The measurement test bed is set up with 100 Mbps LAN connections. Testing the traffic of VoIP gateways requires the use of PSTN phones where the gateway converts voice calls between the PSTN and the IP networks. The tested coders are G.711, G.723.1, and G.729A. The RTP messages of different established calls using each of the available coders are captured and analyzed. The results are summarized in Table 2.2.

Table 2.2: Measured parameters for IP phones and VoIP gateways.

Coder type	**G.711**	**G.723.1**	**G.729A**
RTP payload (bytes)	160	24	30
IP packet size (bytes)	200	64	70
Mean interarrival time (ms)	19.997	30.004	30.084
Standard deviation	0.065	0.348	1.127
Overall rate (IP layer) (kbps)	80.012	17.064	18.615
Number of frames per RTP packet	160	1	3

Referring to Table 2.2, it is easily inferred that all coders generate deterministic traffic with a fixed rate as long as the call is up. G.711 was transmitting a 200-byte IP packet almost every 20 ms time, G.723.1 a 64-byte IP packet every 30 ms time, and G.729A a 70-byte IP packet every 30 ms.

Based on the captured traffic traces, the TB parameters (r, b, p, m, M) for G.711, G.723.1 and G.729A are determined and presented in Table 2.3. At first, we set the token fill rate r to be equal to the long-term average data rate of a traffic flow. Based on this value for r, b can be derived as the minimum token bucket depth, which is needed to avoid packet loss. As we will see in Section 2.3.3, the value of b greatly depends on the token fill rate r. The peak rate p corresponds to the maximum per-packet rate that is observed in the trace. The per-packet rate is calculated as the packet length over the preceding interarrival time. For each coder, two parameter sets are listed. The "ideal" set refers to the deterministic behavior of the coder, which could be observed right at the coder output before transmitting the packets into the IP network. The parameters of this model are calculated knowing that the coders transmit fixed-size packets at constant time intervals. The "measured" parameters refer to the captured traces.

According to the low standard deviation values of the interarrival times, which have been computed in Table 2.2, the measurement results shown in Table 2.3 are all very close to the ideal ones. Only the bucket depth b is almost twice the ideal value for both G.723.1 and G.729A. This can be explained by the fact that the bucket model is specified for zero packet loss and is, thus, considering all packets without exception. Throwing away the few packets, which arrive significantly out-of-profile, can bring down the bucket depth to the ideal value. In any case, the service rates R_{voice} computed for both models are identical due to the fact that the peak rate is almost equal to the mean rate and, thus, the second formula of (2.9) is used. This formula depends only on the M parameter, which is the same in both models. As a consequence, using the measured model would guarantee a zero packet loss and a hard delay limit with no extra capacity cost as compared to the ideal model.

Table 2.3: G.711, G.723.1 and G.729A traffic models for IP phones and VoIP gateways.

Traffic Model						
Coder type	**G.711**		**G.723.1**		**G.729A**	
Model type	**Ideal**	**Measured**	**Ideal**	**Measured**	**Ideal**	**Measured**
r **(kbps)**	80	80.012	17.067	17.064	18.667	18.615
b **(bytes)**	200	213	64	114	70	130
p **(kbps)**	80	84.867	17.067	17.884	18.667	18.716
m **(bytes)**	41	41	44	44	42	42
M **(bytes)**	200	200	64	64	70	70
R_{voice} **(kbps)**	129	129	41	41	45	45

2.3.2.2 Soft Clients

Having soft clients as PC-based phones suggests that the operation of soft clients depends on the manner in which the operating system handles the different running processes. Due to this fact, the performance of soft clients under different levels of priority is tested and two sets of tests are performed between two soft clients, host A and host B. These are

1. *noLoad* test: Host A initiates the call to host B while there is no other application running on the PC of host A.
2. *withLoad* test: Host A initiates the call to host B while the PC of host A is busy copying a 400 Mbytes file from a server connected to host A via a switch.

It was noticed that under unloaded conditions both coders transmit deterministic traffic, as was the case with IP phones and VoIP gateways. However, in loaded conditions, the traffic is non-deterministic leading to dramatic quality degradation in voice calls. Tables 2.4 and 2.5 compare the results of *noLoad* and *withLoad* cases for the two coders G.711 and G.723.1 respectively, which are available in the tested phones. In the *noLoad* case, G.711 transmits a 280-byte IP packet every 30 ms time and G.723.1 a 64-byte IP packet every 30 ms time. However, under loaded conditions with normal priority (*NR withLoad*) the traffic deviates from its deterministic behavior. It is worth mentioning that the load exists at the PC side only and not at the network side since the server is connected to host A via a switch and, thus, the copied file does not flow through the whole network. Examining the traffic from host B to host A supports this idea as it is similar to that of the *noLoad* case.

Table 2.4: Measured parameters for soft clients using G.711 coders.

Test case	**noLoad**	**NR withLoad**	**HI withLoad**	**RT withLoad**
RTP payload (bytes)	240	240	240	240
IP packet size (bytes)	280	280	280	280
Mean interarrival time (ms)	29.997	34.16	33.50	30.06
Standard deviation (ms)	0.185	44.10	44.18	8.16
Overall rate (IP layer) (kbps)	74.674	65.57	66.87	74.52

Table 2.5: Measured parameters for soft clients using G.723.1 coders.

Test case	**noLoad**	**NR withLoad**	**HI withLoad**	**RT withLoad**
RTP payload (bytes)	24	24	24	24
IP packet size (bytes)	64	64	64	64
Mean interarrival time (ms)	29.988	37.6	40.17	29.99
Standard deviation (ms)	0.408	64.1	63.14	5.4
Overall rate (IP layer) (kbps)	17.073	13.61	12.74	17.066

The results of the *NR withLoad* case are explained by knowing that the operating system gets so busy copying a large file from the server. Having no sense of prioritizing the soft client traffic over other traffic, the operating system stops sending the soft client packets for some time while copying the received packets. Then, after interrupting the copying process, the accumulated voice packets are sent out in one burst. This observation is illustrated in Figure 2.6a that plots the rate per packet for G.711 coder computed as the packet size divided by the interarrival time. It is shown that the packet rate changes significantly over time. A similar behavior is observed when G.723.1 is used as manifested in Figure 2.7a.

The same test scenario is then tried by setting the priority level of the soft client process to the HI mode that is an intermediary priority level in the Windows operating system. The same statistics are collected and the obtained results are also shown in Table 2.4 and Table 2.5 for coders G.711 and G.723.1, respectively. However, not a worthy improvement has been observed and the quality stays to be unacceptable. As a result, increasing the priority level of voice applications to the HI mode does not help. Some processes are still receiving preferential treatment by the operating system causing voice packets to be discarded in the PC transmit buffer until a free time slot occurs. In consequence, the *HI withLoad* is not investigated any further and the process priority of the soft client applications is now set to the highest user level called “realtime” (RT) mode.

An application running in realtime mode receives higher priority than all other user processes and, thus, is serviced earlier by the operating system. Performing the same measurements, we note that the performance of the *withLoad* case improved significantly since the operating system is giving priority for the soft client process on the copying process. The extremely bursty traffic rate generated in the *withLoad* case has dramatically been smoothened as compared to *NR withLoad*case. This effect is demonstrated in Figures 2.6b and 2.7b for both G.711 and G.723.1 coders. It is interesting to note that some spikes in the traffic rate still occur every few seconds. This is explained by the fact that some system or kernel processes get an ultimate priority over all user processes and therefore some voice packets still get accumulated in the transmit buffer and then sent directly at once with a high rate to the network.

This improvement is also manifested by sketching the probability mass function (PMF) of the interarrival times before and after starting the RT mode for both coders in Figures 2.8 and 2.9 respectively. It is shown that: in *RT withLoad* case, a high percentage of the packets are being sent in almost 30 ms intervals as opposed to the *NR withLoad* case where packets are sent in a non-deterministic manner with varying interarrival times mostly around 0 ms and 30 ms.

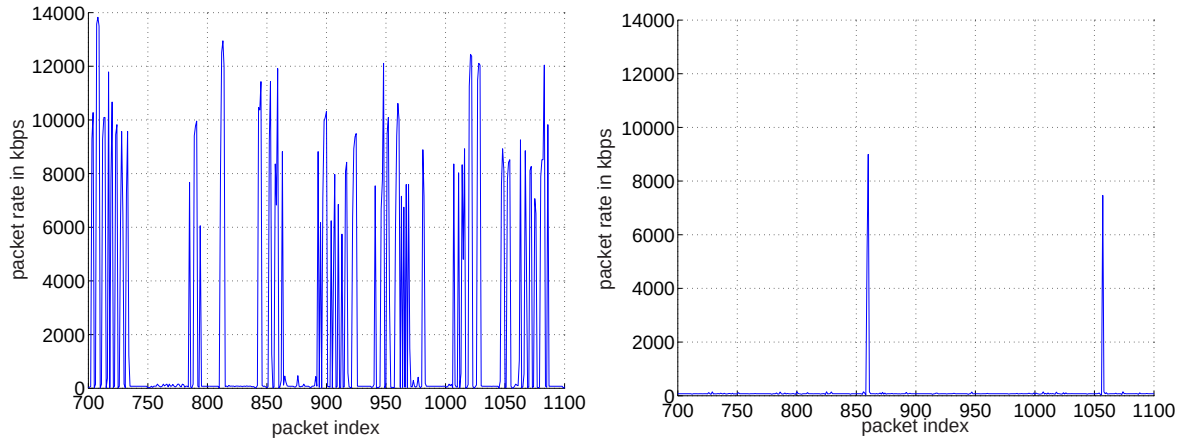


Figure 2.6: Packet rate for G.711 (a) *NR withLoad* case (b) *RT withLoad* case.

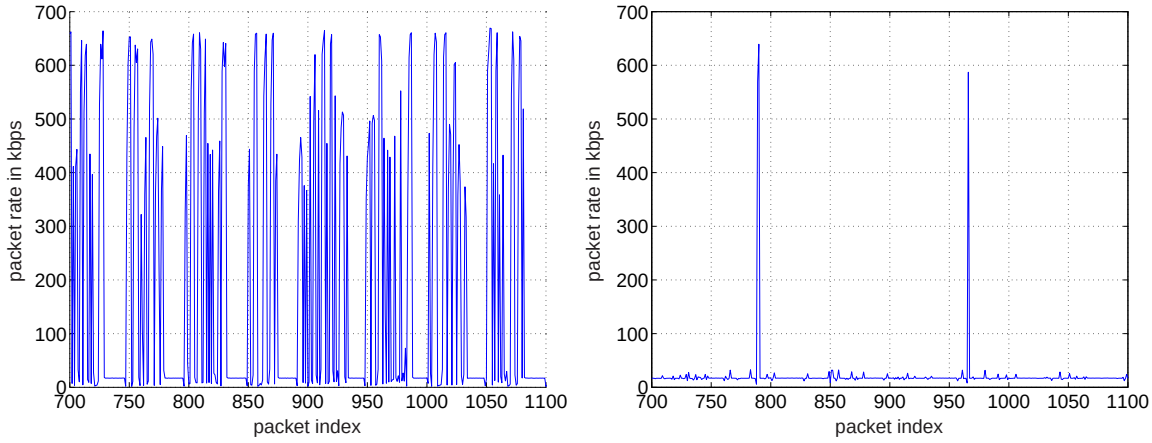


Figure 2.7: Packet rate for G.723.1 (a) *NR withLoad* case (b) *RT withLoad* case.

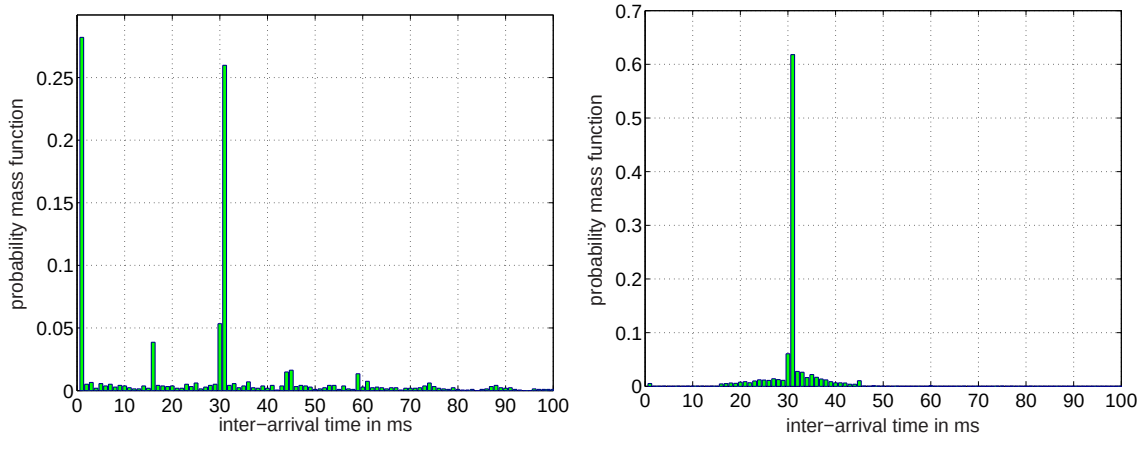
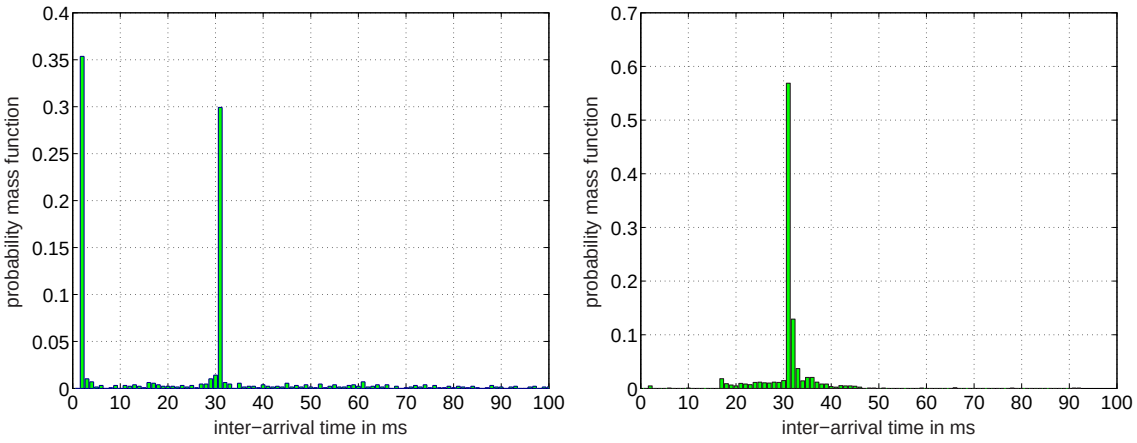


Figure 2.8: PMF of inter-arrival time for G.711 (a) *NR withLoad* case (b) *RT withLoad* case.



(a) (b)

Figure 2.9: PMF of inter-arrival time for G.723.1 (a) *NR withLoad* case (b) *RT withLoad* case.

Using the measured traffic traces, the token bucket parameters of each trace are evaluated and compared to the ideal model in Table 2.6. The ideal and measured token bucket parameters for G.711 and G.723.1 coders are presented for each of *NR withLoad* case and *RT withLoad* case. It is shown that the measured models of the *NR withLoad* case deviate largely from the ideal models in terms of r, b and p. This means that the traffic profile has been extremely distorted, which is not due to the network but due to the operating system of the PC. Concerning the *RT withLoad* case, the token bucket depth b is still much larger than the ideal case. This fact indicates that in spite of prioritization, some packets are still transmitted completely out-of-profile leading to a high bucket depth for zero packet loss. However, it is expected that by accepting these rare packets to be lost would reduce the bucket depth significantly.

Table 2.6 shows that the service rate R_{voice} computed for G.711 coder in *RT withLoad* is remarkably smaller than the one in *NR withLoad*; yet, it is almost four times the ideal R_{voice}. In the next section we will discuss possibilities to reduce this rate by tuning the token bucket description parameters on one hand and by allowing a few packets to be lost on the other hand.

Table 2.6: G.711 and G.723.1 traffic models for soft clients.

Traffic Model								
Coder type	**G.711**				**G.723.1**			
Model type	**Ideal**	**Measured**			**Ideal**	**Measured**		
		noLoad	**withLoad**			**noLoad**	*withLoad*	
			NR	**RT**			**NR**	**RT**
r **(kbps)**	74.667	74.67	65.57	74.52	17.067	17.073	13.61	17.066
b **(bytes)**	280	370	12453	3774	64	95	4235	232
p **(kbps)**	74.667	84.848	14545	12948	17.067	20.539	673.7	639.2
m **(bytes)**	41	41	41	41	44	44	44	44
M **(bytes)**	280	280	280	280	64	64	64	64
R_{voice} **(kbps)**	180.5	180.5	1894	715	41	41	360	66

2.3.3 Token Bucket Parameter Trade-offs and Effects on Capacity Requirements

The TB parameters in the preceding sections were computed by setting the token fill rate equal to the respective long-term average data rate. As we have seen, this approach works fine as long as the traffic shows almost constant bit rate behavior. However, for the cases where bursts arise due to processing loads on the PCs of softclients, the bucket depth b and, consequently, the service rate R_{voice} become quite large. In this section, we look at two possibilities to reduce this service rate R_{voice} and, thus, to minimize the capacity shares that have to be reserved within the network. It is important to note that the service rate that needs to be provided within the network is determined based on worst-case considerations. Therefore, by adjusting the TB parameters, the worst-case behavior of a traffic flow passing through the TB can be limited, leading to capacity savings in the network. Principally, there are two ways of reducing the required service rate R_{voice}. These are

1. increasing the token fill rate leading to smaller bucket depths,
2. introducing packet losses resulting in attenuated bursts.

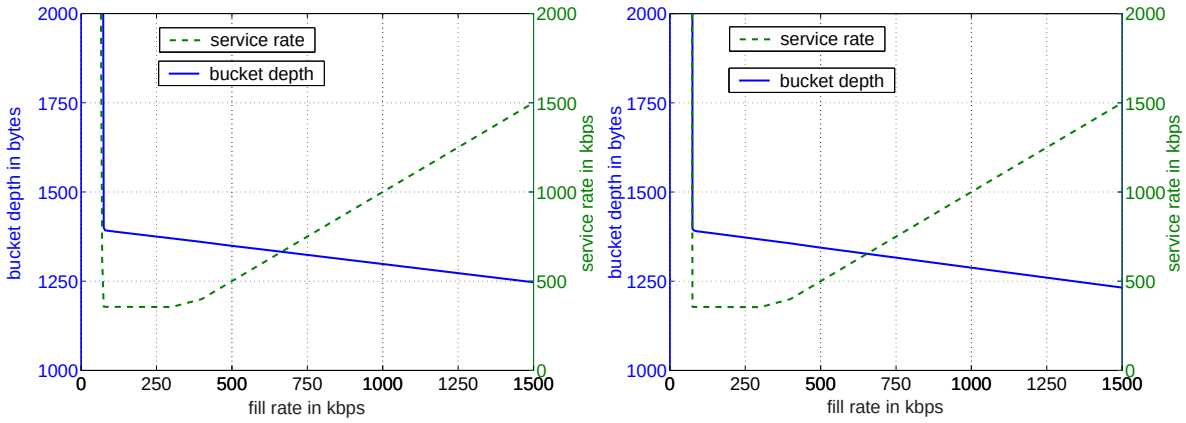
2.3.3.1 Service Rate vs. Token Fill Rate

At this point we still demand that no packets are lost due to the token bucket policer (i.e. $A^* = A$). However, instead of using the average data rate as the token fill rate r, we now increase r and observe the effect on the resulting service rate R_{voice}. The token fill rate r directly influences the bucket depth b. Having a higher r allows us to reduce b since bursts, which arise due to long-term deviations from the mean rate, are shortened. As r approaches p, the bucket depth would go down to its ideal value. As b decreases, the worst-case burst becomes more and more limited, allowing the service rate R_{voice}, which needs to be reserved within the network, to be decreased, too.

These two effects are demonstrated in Figure 2.10, which shows the trade-off behavior for G.711 in the *NR withLoad* case. Starting with the original token fill rate of around 65 kbps would require a bucket depth of roughly 12500 bytes and a service rate of around 2 Mbps. Increasing r only a little bit brings down these values to 1400 bytes and 400 kbps, respectively. A similar behavior is observed for the *RT withLoad* case. However, if r is firstly set to 65 kbps as in the *NR withLoad* case, a huge bucket depth of around 150 000 bytes is resulted, which is already 12 times as large as the depth required in *NR withLoad* case when r is set to the same value. The reason is that the generated traffic in *RT withLoad* case has an overall rate of 74.52 kbps which exceeds the used value of 65 kbps. In order to compensate for the low r, a huge b is required so that no packet is lost. Whereas, if r is set to the actual mean rate (i.e. 74.52 kbps), 3774 bytes are required as the bucket depth leading to a reduced service rate requirement of $R_{\text{voice}} = 715$ kbps. A considerable reduction in R_{voice} is further achieved when a slight increase in r is imposed. For $r \geq 75$ kbps, R_{voice} drops below 400 kbps. As r grows larger, the service rate reaches a point where it increases again; this is due to the fact that R_{voice} must be greater than r at all times. The minimum R_{voice} achieved evaluates to 355 kbps. It is interesting to note that for a certain value of r, both *NR withLoad* and *RT withLoad* cases lead to almost the same service rate. This is logical as both cases generate extremely high spikes in the packet rate (refer to Figure 2.6) and the TB is designed in a way to accommodate all generated packets.

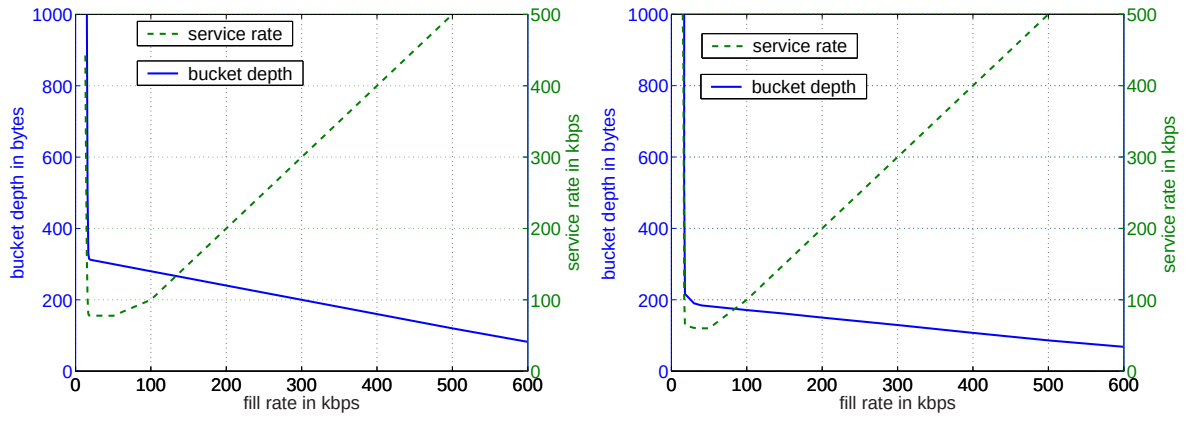
Therefore, we reach a point where for some r equal bucket depths are required in both cases –*NR withLoad* and *RT withLoad*– in order to accept all packets caused by the high spikes no matter how often these spikes occur leading finally to similar R_{voice}.

Figure 2.11 presents the trade-off behavior for G.723.1 coder that is characterized similarly as for G.711 coder. We can say also that a slight increase in r brings down the bucket depth significantly and so the required service rate. However, after r exceeds some value, the bucket depth is further reduced while the service rate increases proportionally to the increase in r. The minimum achieved value of R_{voice} in both cases amounts to 60 kbps.



(a) (b)

Figure 2.10: Bucket depth and service rate versus token fill rate for G.711 (the bucket depth in both cases is initially very huge but the maximum bucket depth shown in the figure is 2000 bytes in order to show finer granularity in the behavior of the curves) (a) *NR withLoad* case (b) *RT withLoad* case.



(a) (b)

Figure 2.11: Bucket depth and service rate versus token fill rate for G.723.1 (a) *NR withLoad* case (b) *RT withLoad* case.

2.3.3.2 Service Rate vs. Packet Loss

As could be seen in Section 2.3.3.1, the service rate can be reduced by adjusting the token bucket parameters in a way that they tightly fit the traffic characteristics. Nevertheless, the minimum service rates of 355 kbps and 60 kbps for G.711 and G.723.1, respectively, are still quite high, knowing that the average rate is much lower. This is due to a few extreme bursts, which require high service rates if no packets should be lost. However, by allowing packet losses we can thin out the long bursts and, thus, be able to reduce the bucket depths as well as the service rates. The desired packet loss can be realized by the TB policer if dimensioned properly.

Let us first consider a TB, which is dimensioned according to the ideal, constant-bit-rate model and which, therefore, leads to packet losses. Table 2.7 shows that a packet loss of less than 5% is obtained in IP phones, VoIP gateways, and *noLoad* soft client cases. That amount of packet loss seems to be acceptable to maintain a good voice quality. However, soft clients in loaded situations suffer from extremely high packet loss ranging from 25% to 50% causing impossible interaction between the communicating parties. One might argue that the *NR withLoad* case would anyway lead to terrible communication quality even if no packets are lost and this is due to the high delays experienced by a major number of the voice packets which are queued in the PC transmit buffer for already a time exceeding the acceptable limit. So, in anyways whether a packet loss is introduced or not, the quality is not suitable for interactive communication and the communicating parties would most likely hang off the call. Otherwise, they perhaps stop the demanding process (e.g. the copying process) running in parallel and thus the situation is classified as *noLoad*. The problem lies in the *RT withLoad* case which provides a desirable communication quality in spite of the very rare spikes that occur in the packet rate: a packet loss above 25% would definitely worsen the communication quality in a significant way causing the call to be dropped. However, increasing the bucket depth to twice its ideal value already achieves huge performance gains where the packet loss is dropped to less than 0.5%. We note also that a bucket depth of twice the ideal value ($2 \times \text{ideal}(b)$) leads to satisfactory performance values for all other client situations. This is illustrated in Table 2.7.

Table 2.7: Resulted packet loss using defined TB models.

Case description	Packet loss			
	G.711		G.723.1	
	ideal(b)	2×ideal(b)	ideal(b)	2×ideal(b)
IP phones	$< 5\%$	0%	$< 5\%$	0%
VoIP gateways	$< 5\%$	0%	$< 5\%$	0%
noLoad soft clients	$< 5\%$	0%	$< 5\%$	0%
NR withLoad soft clients	52%	25%	55%	27%
RT withLoad soft clients	41%	0.5%	24%	0.5%

For further investigation of soft clients under loaded situations, we examine the bucket depth and the corresponding service rate in terms of the induced packet loss. In *RT withLoad* using G.711, a service rate of 715 kbps would be needed for zero packet loss, while just 180 kbps is required by a normal deterministic traffic to obtain high quality (refer to Table 2.6). Dimensioning for zero packet loss would then be unreasonable. Allowing very few packets to be lost (i.e. cutting off the last packets of a burst) would lead to satisfactory results.

Figures 2.12 and 2.13 show that a slight increase in packet loss in the *RT withLoad* case reduces the bucket depth significantly, and thus, the service rate. On the other hand, a high loss is required in *NR withLoad* case to achieve a considerable reduction in the bucket depth and consequently in the service rate. The reason for this is that bursts appear very rarely in *RT withLoad* case and so few packets have to be dropped in order to achieve a more regular packet flow. In *NR withLoad* case, many bursts exist, which all have to be affected to smoothen the traffic.

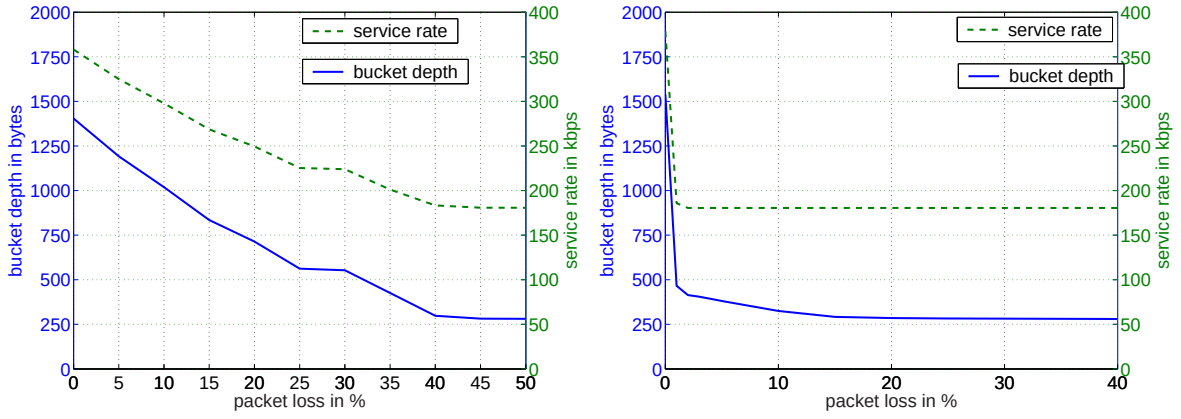


Figure 2.12: Bucket depth and service rate versus packet loss for G.711 (a) *NR withLoad* case (b) *RT withLoad* case.

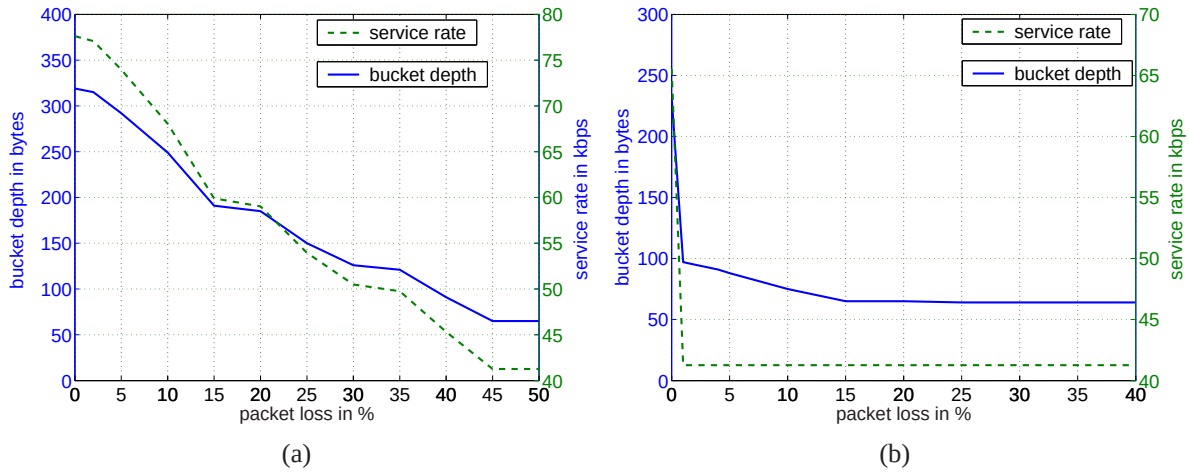


Figure 2.13: Bucket depth and service rate versus packet loss for G.723.1 (a) *NR withLoad* case (b) *RT withLoad* case.

2.4 Interactive Video Services

In this section, we focus on videoconferencing applications and aim for examining their traffic characteristics and capacity requirements based on a similar methodology to that applied to voice over IP applications. In this regard, we refer to [Gla03] [EG04] where extensive measurements of such applications are performed under various conditions. Statistics are also collected from measured traffic traces to evaluate the token bucket parameters. In order to evaluate the capacity requirements for individual flows, we use the token bucket parameters obtained in [Gla03] and apply them into (2.9) such that the given network delay bound is never exceeded providing hard QoS guarantees. We assume similar performance and system parameters as in Section 2.3 where a video communication is established between two users separated by three routers using WFQ scheduling and links of speed 100 Mbps. The link MTU is assumed to be 1500 bytes as in Ethernet. The required network delay bound is given as $\widehat{\mathcal{D}}_{\text{net}}$ = 50 ms, and propagation delay is ignored.

2.4.1 Description of Video over IP Clients

The publicly-used videoconferencing applications are mostly software-based. Some do still require additional hardware such as stand-alone DSPs (digital signal processing). The applications tested in [Gla03] are commercial applications known as LiveLan, Vcon, and Netmeeting. The former two require a separate DSP and the latter is purely software. In order to produce reliable and accurate traffic characteristics for the applications at hand, background traffic is avoided here as much as possible and is being monitored whether it exceeds a certain load limit. Table 2.8 presents the different videoconferencing applications used for the measurements and compares them among each other.

Table 2.8: Description of the measured videoconferencing applications. CIF stands for common intermediate format and QCIF for quarter CIF.

Application	**LiveLan**		**VCON**		**Netmeeting**		
Hardware	analog camera, PCI card		digital camera, PCI card		digital camera, parallel port		
Medium	Video	Audio	Video	Audio	Video	Audio	
Settings	64–768 kbps		64–768 kbps		picture size	–	
	QCIF, CIF	–	QCIF, CIF	–	quality		
Coder type	H.261	G.711	H.261	G.728	proprietary	G.723.1	G.711
Quantization (bit per pixel or sample)	24	8	24	8	unassigned	16	8
Compression factor	100	1	100	4	unassigned	20.31	1
Frame size	$<$ 32 Kbyte	8 bit	$<$ 32 Kbyte	10 bit	unassigned	189 bit	8 bit
Frame rate (Hz)	7.5–30	8000	7.5–30	1600	unassigned	33	8000
Peak rate (kbps)	7758	64	7758	16	unassigned	6.4	64
Coder rate (kbps)	110–704	64	48–752	16	unassigned	≤ 6.4	64

All tested applications employ audio and video coders. The audio coders are constrained with constant upper limit on the transmission rate while video coders have a range of transmission rate which depends heavily on the selected settings and the movement behavior of the users. The settings of LiveLan and VCON are configured by the picture size (QCIF: 176×144 pixel, CIF: 352×288 pixel) and the transmission rate r_{set} that includes both video and audio data

streams. This overall transmission rate ranges from 64 kbps (voice alone) up to 768 kbps (voice and video). In case this rate is exceeded, the application reduces the quality of the picture in terms of quantization and resolution. Netmeeting, on the other hand, does not offer such setting possibilities neither does it offer the selection of the desired video coders. Instead, Netmeeting allows to specify qualitative values for the picture size and the quality. In regards to the video coder, it is automatically set by the application itself depending on the given settings. Even when the RTP headers of the exchanged messages are examined to check for the coder in use, the *PayloadType* field of the header is set to "unassigned". In the following, we briefly present the most common video coders.

2.4.1.1 Video Coders

Due to the extremely high capacity requirements of uncompressed video data streams, several compression algorithms have been developed to reduce the amount of data needed to be transmitted, yet maintaining an acceptable level of quality. An uncompressed HDTV (high definition television) picture with 2.2 million pixels and raw coding with 24 bits per pixel, for example, would require 1.5–3 Gbps depending on the picture frequency. Obviously, sophisticated coding is required to cut down the capacity requirements to few Mbps. There exist several coding methods in the literature, which are used to reduce the amount of transmitted information. These methods form the basis of the coding standards such as JPEG (Joint Photographic Experts Group), MPEG (Moving Picture Experts Group), H.261, and H.263.

The available coding standards permit both constant and variable bit rate transmission [RMV96]. Due to the varying picture content of video transmission, the rate is naturally variable. Constant rate transmission is obtained when a closed-loop control is used to adjust certain coder parameters to feed a transmit buffer with sufficient data, maintaining the constant output rate. Constant-rate encoding leads to easy-to-handle transmission but at the cost of varying visual quality and increased packet latency which can be critical for interactive realtime applications. The increased packet latency is due to the delay introduced by the smoothing transmit buffer, which is needed to provide the constant output rate. The open-loop control, however, provides a stable quality but at the cost of network planning and management complexity. It allows for wide range of variation of the output rate depending on the scene filmed, thus, making the transmission rate and the network delay more difficult to predict. A compromise solution is to allow variable bit rate coding with controlled transmission. This is achieved by means of a loose control loop where output variability is limited with a token bucket that shapes the traffic output to satisfy its configured parameters.

Several standards have been developed throughout the years for video transmission mainly by the MPEG working group and the ITU-T in Recommendation H.261 and H.263. Below is a brief description of the most common and standardized video coders.

MPEG-1 –specified in [ISO91]– has become the "de facto" standard. It was designed for a target bit rate of 1.5 Mbps. The typical image format is CIF (common intermediate format) with no interlace. It generates frames at a rate ranging from 24 to 30 fps (frames per second). MPEG-1 was originally created for video storage for multimedia on compact discs. It is extremely popular in the form of video CDs. In addition, level 3 of MPEG-1 is the most popular standard for digital compression of audio –known as MP3.

MPEG-2 –specified in [ISO94]– is based on MPEG-1 but is capable of achieving higher compression ratios and of supporting interlaced videos. It is a generic coding standard which is

optimized for TV resolution and can be used for stored video and realtime applications. It is designed for the compression and transmission of digital broadcast television. MPEG-2 scales well to HDTV resolution obviating the need for an MPEG-3.

MPEG-4 –specified in [ISO99]– is based on object-based compression. It is a standard for a wide range of applications with choices of interactivity, scalability, error resilience, and others. These features are achieved by the fact that individual objects within a scene are tracked separately. They can be compressed together allowing for high compression efficiency and can independently be controlled in a scene allowing for high interactivity.

MPEG-7 –specified in [ISO01]– is also called the multimedia content description interface. Unlike the previous MPEG standards which describe the content, MPEG-7 provides information about the content. It is designed to provide a generic framework for multimedia content that includes information on content manipulation, filtering, personalization as well as security.

H.261 –specified in [ITU93]– is designed for two-way communications over ISDN links and supports low data rates at multiples of 64 kbps up to 1.92 Mbps. It is also designed for low coding delay making it suitable for interactive realtime communications. H.261 is based on discrete cosine transform and supports a frame resolution of QCIF (quarter CIF) and CIF. It uses intra-frame and inter-frame compression.

H.263 –specified in [ITU96f]– is based on H.261 but comprises enhancements in motion compensation, quantization, and variable-length encoding. Although it is designed for very low bit rates starting at 10 kbps, H.263 performs quite well at higher rates up to 2 Mbps. It supports frame resolutions of CIF, QCIF, sub-QCIF, 4CIF, and 16CIF. H.263 has been superseded by the newer version H.263+ –specified in [ITU98c]. H.263+ contains further improvements especially in compression efficiency, error-resilience, and bit stream scalability.

2.4.2 Measured Traffic Characteristics of Video over IP Clients

The available videoconferencing applications are started and a network monitoring tool is used to capture the RTP messages exchanged between the two communicating parties. In this case, tcpdump [Tcp01] and ksnuffle [Ksn01] are used. RTP messages are saved in files which are processed afterwards to compute the token bucket parameters. We note that only one-to-one communications are performed, though results can be applied in a straightforward manner to point-to-multipoint communications.

[Gla03] differentiates among three kinds of peak rate that are p_{z} , p_{q}, and p_{b}. The former one p_{z} refers to the actual peak rate of the trace, i.e. the maximum packet rate. The packet rate is computed as the packet size divided by the interarrival time between the given packet and the preceding one. However, it can happen that occasional frames with a size exceeding one MTU are generated and are thus partitioned into smaller IP packets. These packets are then sent out together at once into the network as a burst causing a high peak rate. When such high rates occur very rarely in the transmission process, then it might be due to some network disturbance or an exceptional behavior of the user. As a result, these effects are eliminated by using the 99.9% quantile of all sample packet rates. The 99.9% quantile value is referred to as p_{q}. Referring to Table 2.9, we realize that the peak rate of the LiveLan application at $r_{\mathrm{set}} = 384$ kbps can be reduced from 9.9 Mbps to 2.0 Mbps and at $r_{\mathrm{set}} = 174$ kbps from 9.9 Mbps to 0.4 Mbps if one uses p_{q} instead of p_{z}. A huge reduction in the peak rate is also achieved in the Netmeeting application at “medium” picture size and “fast” quality settings. However, in case high packet

rates occur very frequently due to bursts resulting mainly from large video frames, then using p_{q} would not help as illustrated in Table 2.9. p_{q} continues to be as high as p_{z} and equal to 9.9 Mbps for the Vcon application at 384 kbps settings even if the 0.1% of high peak rates are discarded. In such cases, the peak rate should be computed differently so as to achieve lower values. It is computed by adding up the sizes of all packets belonging to one burst and dividing them by the burst duration. The resulting peak rate is referred to as p_{b}.

Table 2.9 summarizes the results for each of the tested applications in different settings. Several measurements are performed for different r_{set} values. In the *Settings* column of Table 2.9, *P* and *Q* refer to picture size and quality respectively. The token bucket parameters are estimated for a sample individual video flow conforming to the indicated settings, and the highest value is selected for each of the parameters so as to assure conservative results. As for the peak rate, three values are estimated corresponding to p_{z}, p_{q}, and p_{b} respectively.

Based on the token bucket and the given system parameters, we compute the service rate required for one flow to achieve a deterministic quality of service in terms of network latency using (2.9). We note that three values for the service rate are computed, each of which corresponds to a different peak rate indicated at the top of each subcolumn in *Service rate* column of the table. It is obvious that p_{z} is higher than p_{q} in all cases; however, the difference ranges widely from small values up to large ones depending on the occurrence frequency of the bursts associated with the high rates. By reducing the peak rate to p_{q}, the required service rate is consequently reduced as it is evident in Table 2.9. However, we note that only a slight reduction in the service rate is achieved except in cases when the peak rate drops tremendously. Using p_{b}, the required service rate drops in average by almost 39%. It is also observed that the computed fill rate value is comparable to the set value r_{set}. In [Gla03] [EG04], a detailed analysis is available related to the required service rate and its behavior in terms of several parameters including the token fill rate, the number of hops along the communications path and the tolerated packet loss.

Table 2.9: Traffic models for Video over IP clients.

Video Application	**Settings**	**Token bucket**						**Service rate $\mathbf{R}_{\mathrm{video}}$, kbps**		
	r_{set} **kbps**	r **kbps**	b **kbyte**	p_{z} **Mbps**	p_{q} **Mbps**	p_{b} **Mbps**	M **bytes**	p_{z}	p_{q}	p_{b}
LiveLan	768	750	9.0	9.9	9.5	2.0	1520	2018	2011	1481
	384	350	7.0	9.9	2.0	1.0	1520	1735	1335	991
	174	120	2.5	9.9	0.4	0.4	1520	1122	771	771
Vcon	384	400	8.0	9.9	9.9	0.9	1470	1840	1840	915
	128	120	3.2	9.9	6.2	0.4	1470	1196	1178	674
Netmeeting	P:medium Q:best	350	8.0	9.9	9.9	0.8	1430	1818	1818	836
	P:medium Q:medium	100	2.5	9.9	9.9	0.4	1430	1077	1077	731
	P:medium Q:fast	70	1.8	9.9	0.9	0.2	1430	976	920	695

2.5 Summary

In this chapter, we determined traffic characteristics for individual traffic flows of different multimedia applications through monitoring actual calls in different situations. The token bucket parameters have been utilized to quantitatively describe the characteristics and to calculate the corresponding capacity requirements for each flow. While VoIP traffic is usually quite deterministic and pretty smooth, it can happen that soft client applications running under highly loaded operating systems inject flows, which are very "out-of-profile". If these traffic flows should still be carried by the network with the desired QoS, the service rates have to be set appropriately leading to extremely high capacity requirements. However, the voice quality of these calls is not acceptable anyway and the call will most likely be dropped. This fact motivated the need for running soft client applications in realtime mode so they receive preferential treatment by the operating system over other parallel-running applications. Doing so, the call quality improved notably and the service rate was extensively reduced as well, yet still being several times higher than the rate required by an ideal flow. This is due to some occasional bursts that are generated in a seldom manner but are still accounted for so as to achieve zero packet loss. Should these rare bursts be discarded, the service rate drops down tremendously to acceptable values. Moreover, we presented possibilities how adjusting the token bucket parameters can further reduce the necessary service rate.

Analogous analysis is also done for different video clients which are tested in different settings in [Gla03] [EG04]. We presented the token bucket parameters results obtained for three commercial videoconferencing applications with different settings. In the case of video transmission, traffic peak rate can be extremely high due to large video frames that are partitioned into smaller packets to fit within an MTU packet and sent at once to the network. In the light of this, three different ways for computing the peak rate are proposed in the cited work and a detailed analysis is performed to obtain the token bucket parameters leading to the most economical service rate and yet achieving a high quality level.

Finally, we conclude that using a token bucket policer per each flow at the network ingress of our network model assures that incoming flows of multimedia applications including both voice and video behave smoothly and conform to their given profile. Individual conforming flows would require acceptable but "multiples-of-mean-rate" service rates for hardly-guaranteed QoS. With deterministic aggregation of the active flows and their capacity requirements, no multiplexing gain is obtained leading to large waste of resources for higher number of active flows. This motivates the need for a rather statistical method that evaluates the capacity needs of a given aggregate of flows for high quality. In the next chapters, we address this important issue and work for new capacity assignment methods that consider the inherent characteristics of voice and video communications over IP networks.

3

Network Dimensioning for Voice Services with Statistical QoS Guarantees

We have observed in Chapter 2 that nicely-behaving voice over IP sources not incorporating VAD generate deterministic traffic: a constant packet size is transmitted every fixed interval of time. It might happen that some voice sources generate bursty traffic when operating under certain situations like soft clients for example. The task of traffic policers placed at the output of each source assures that the generated traffic fits to a predetermined profile otherwise it is shaped to fit to this profile. Based on this given profile, the capacity requirement of each conforming flow is independently evaluated from other flows for a deterministic quality of service. This leads to a relatively high capacity as illustrated in Chapter 2. In this chapter, we aim at evaluating the capacity requirement for an aggregate of VoIP flows at the minimum network cost while still achieving a premium quality level.

This chapter is structured as follows. In Section 3.1 and Section 3.2, we apply an analysis-synthesis approach on a general network model to decompose the network into its constituent elements at various levels –network level, path level, node level and buffer level. We examine the properties of the constituent elements and then compose them into the general network model again going through the same levels and taking into account the different interdependencies that exist from one level to the other. In Section 3.3, we investigate various capacity assignment strategies at the basic detail level –namely the buffer level– and assess the performance and drawback of each. Motivated by the results of Section 3.3, we define a novel capacity assignment method in Section 3.4 that provides satisfactory tradeoff performance. We provide mathematical analysis of the new method at the buffer level and extend it gradually to be applicable at the node and path levels and finally at the network level where optimization techniques are used. In Section 3.5, the method is finally realized into a prototypical network planning tool with a generic layered architecture that easily accommodates different service types.

3.1 Analysis-Synthesis Approach

Any stand-alone object can be complex in nature and can basically be decomposed into simpler molecular ingredients at different abstract levels. To examine one functionality of the whole object, a detailed study of the basic ingredients at the deepest detail level of interest can be extremely helpful. Specifically, this approach helps in clearly identifying the relevant properties of the basic ingredients and then building up on these properties by including the effect of other ingredients from the higher level, up until the whole object is totally composed again. This procedure represents the basic analysis-synthesis approach. In this chapter, we exploit this basic ideology for the purpose of determining the capacity requirements of the voice service in a given network. In Figure 3.1, we apply the analysis-synthesis procedure on a general network model and present the different detail levels starting from the network level down until the buffer level, going through the path and node levels.

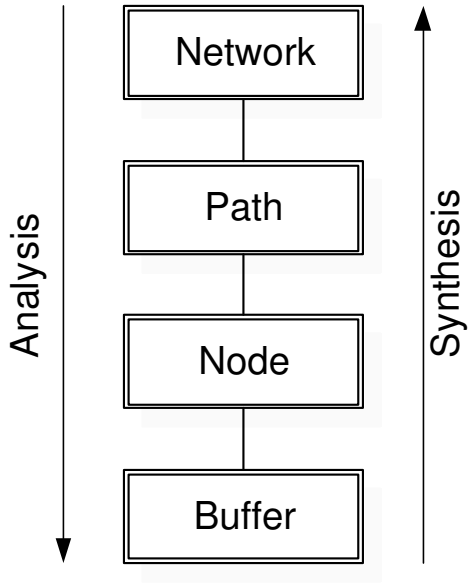


Figure 3.1: Analysis-synthesis approach applied on a communication network.

Figure 3.2 shows the models observed at each of the detail levels. Starting with the general network model composed of nodes and links, we zoom in to observe a set of many communication paths through which information transmission among origin-destination pairs takes place. Further zoom shows each and every communication path as consisting of a succession of nodes in tandem. Nevertheless, each node is composed of a set of buffers or queues each of which is associated to one traffic class (in DiffServ-enabled network for example). As we are basically interested in the capacity requirements of the voice service in the network, the operation of the voice queue is specifically important for our investigation. If the operation is identified, we can proceed on to the next upper levels to account for various factors that affect this operation. In the following section, we provide a brief description of the various models presented in Figure 3.2 at their different detail levels along with the assumptions taken at each level.

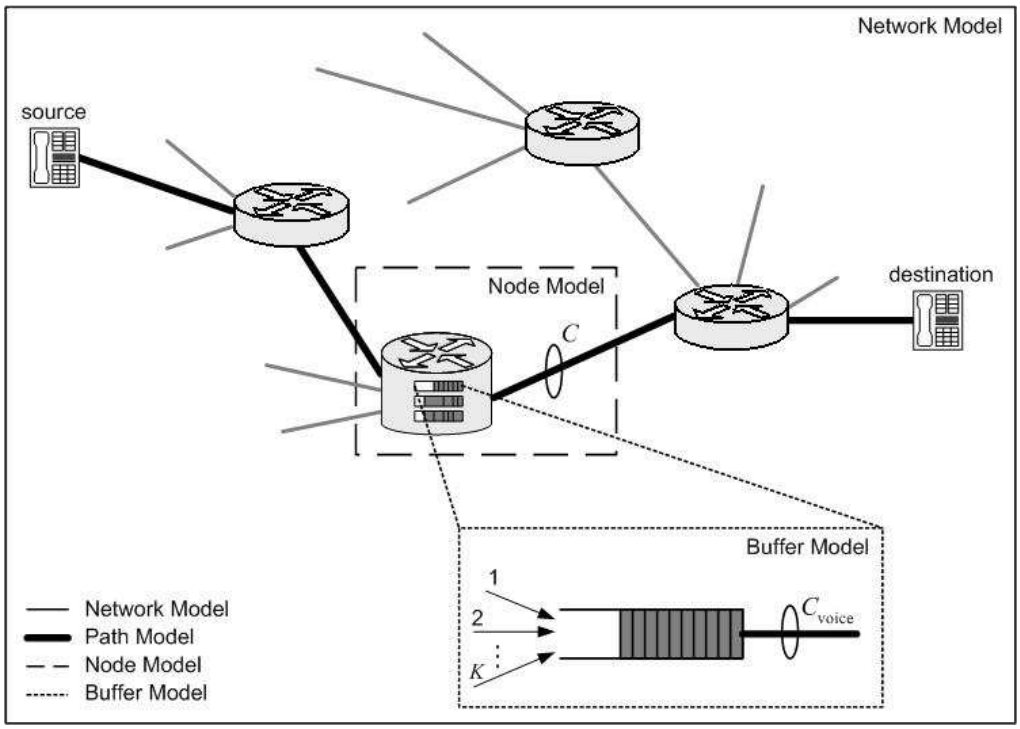


Figure 3.2: System models at different detail levels.

3.2 System Models and Assumptions

3.2.1 Network Level

We consider a general IP network model consisting of a number of nodes and links, through which traffic with different characteristics and requirements could be flowing. Traffic sharing common performance criteria is grouped into one service class supported by the network technology. With multiple traffic classes, the network is attributed as multiservice where a differentiated treatment among the available classes is provided. We assume our network model as supporting a wide range of services including data, voice, and video where each is allocated a separate traffic class. At this point, we note that we are concerned with capacity evaluation for the interactive voice service.

For voice over IP to be widely accepted, it will have to provide a quality level comparable to conventional telephony systems. One crucial factor affecting the perceived quality of voice communications is the end-to-end delay of voice samples exchanged between the speaker and the listener. In VoIP systems, this delay consists mainly of encoding and decoding delays as well as packet transfer time from the sender to the receiver. Furthermore, in order to equalize jitter (delay variation), which arises from traffic variations within the network and change in network state, a playback buffer is employed at the receiver side. Incoming packets whose interarrival times might vary are temporarily stored and played out in equidistant time intervals just as they were sent out by the original sender. This playback buffer time also contributes to the overall delay budget. Altogether, the one-way end-to-end delay value should be less than 150 ms to allow for interactive communications [ITU00].

Having a rather constant coder and playback buffer delay, one can derive a certain maximum threshold for the network delay. Thus, for IP networks to be able to appropriately support VoIP services, network latency has to be kept lower than this threshold. We denote the network delay threshold as $\widehat{\mathcal{D}}_{\text{net}}$ and it represents the maximum delay allowed for any voice packet to traverse the network from one end to the other regardless of the path it uses.

3.2.2 Path Level

Information transmission between a pair of nodes forms communication paths that traverse multiples of nodes in succession. In Figure 3.2, a highlighted voice flow from source to destination is depicted. A flow is assumed to follow one path though results can be extended in a straightforward manner to account for multipath routing. Quality of service constraints of the voice flow at hand are directly related to the delay threshold from source to destination. This delay threshold is denoted as $\widehat{\mathcal{D}}_{\text{path}}$ to differentiate it from $\widehat{\mathcal{D}}_{\text{net}}$. However, we note that $\widehat{\mathcal{D}}_{\text{path}}$ can typically be equivalent to $\widehat{\mathcal{D}}_{\text{net}}$ except that $\widehat{\mathcal{D}}_{\text{net}} = \max(\widehat{\mathcal{D}}_{\text{path}})$ for all available paths in the network. Should we be conservative, we consider a pessimistic scenario in which traffic interfering with the target flow is injected at each node independently from other nodes so as to cause different interference at the different nodes [MWKB99] [KT01] [Kle75]. Consequently, the delay caused by the interfering traffic is independent at each node, allowing us to investigate the per-hop delay contributions autonomously as done in the following section.

3.2.3 Node Level

Based on the assumption of independent hops, we extract a single hop from the path of the target flow and analyze the multiplexing process of several incoming lines onto one outgoing port of the associated node as shown in Figure 3.2. We assume that K active voice connections are carried by those incoming lines and they are destined to the same output link with capacity C. To obtain a high-level performance of the traffic classes supported in the network node considered, sufficient amount of capacity C should be provided to be shared among all classes. As voice traffic demands strict QoS requirements, its share of the link capacity should be explicitly evaluated. Later in the chapter, we present and investigate a new method to compute the voice share of the link capacity, which is denoted as C_{voice}. We define C_{voice} as the least capacity requirement for voice traffic class so it can be served with the desired quality level. The relation between C_{voice} and the whole link capacity is highly dependent on the type of scheduling policy employed in the network node. The scheduling policy defines the way in which packets belonging to the different traffic classes are served. According to a configured way of service, one can apply the differentiated-services treatment in a multiservice network.

In this work, we consider two common scheduling policies, namely priority queuing (PQ) and class-based weighted fair queuing (CB-WFQ). PQ assigns priority levels to the different classes. Packets in a lower priority class are not processed until all packets of higher priority classes are served and the corresponding output buffers are empty. A preemptive type of PQ aborts transmission of a lower priority packet upon the arrival of a higher priority one, whereas a non-preemptive type allows the completion of the transmission. PQ has been proposed as an adequate scheduling for the expedited forwarding per-hop behavior (EF-PHB) [JNP00], which grants premium service to a defined aggregate of traffic in the DiffServ model [BBC98] and has been introduced to support critical realtime traffic such as voice. With PQ, however, it is

possible that traffic classes with a high priority take up the whole bandwidth and push out lower-priority traffic unless some sort of traffic policers is employed to control the high-priority traffic. An alternative to PQ is CB-WFQ. It allocates a weight to each class and shares the link capacity among the different classes that have non-zero traffic volume at the node in direct proportion to their assigned weights. Thus, no traffic class is capable of seizing the whole link at congestion times. We note that both scheduling schemes are work-conserving and so idle periods of one traffic class can be used by another class. If PQ is employed and voice traffic is set to the highest priority as it is very time-sensitive, C_{voice} refers to the minimum link capacity while if CB-WFQ is employed and voice traffic is granted a bounded percentage of the link capacity, C_{voice} refers to the capacity share to be allocated for voice traffic only.

At node level, quality of service is directly related to per-hop delay threshold named $\widehat{\mathcal{D}}_{\text{link}}$ that bounds the different types of latency occurring in one network node and its associated link including processing delay and propagation delay.

3.2.4 Buffer Level

Due to the stringent QoS requirements and particular traffic characteristics of voice service (low data rate and low burstiness), voice traffic is independently treated from other traffic in the network as one traffic class by allocating a separate queue per output port for all voice traffic arriving to a network node [KT00] [KT01]. Figure 3.2 shows the designated voice queue that is depicted in the buffer model.

For simplification, we assume that the K voice traffic flows of the active voice connections are carried by different incoming lines. We need to make sure via proper capacity allocation that the queuing delay of voice packets is bounded within a delay threshold $\widehat{\mathcal{D}}_{\text{buf}}$ in order to attain a smooth and interactive communication between the users. $\widehat{\mathcal{D}}_{\text{buf}}$ is related to $\widehat{\mathcal{D}}_{\text{link}}$ by the fact that $\widehat{\mathcal{D}}_{\text{buf}}$ represents the delay threshold at the buffer level without accounting for the residual transmission time of other traffic classes that are assigned to different buffers.

Therefore, we can represent our problem as a queuing system whose performance depends on the arrival and service processes of voice packets. The arrival process depends on the pattern in which traffic flows are generated by the voice sources and the service process depends on the traffic characteristics in addition to the service mechanism or scheduling policy employed in the multiclass-network node.

Voice sources transmit fixed-size packets at regular intervals if voice activity detection (VAD) is not employed. In case VAD is activated in the voice encoders of the sources, it is expected that the resulting waiting time distribution is bounded by the distribution of the non-VAD-traffic. So, if the latter distribution (that of non-VAD-traffic) is used for capacity allocation then additional overdimensioning for voice traffic class is achieved. According to [KT01] [SW86], it is shown, however, that very slight overdimensioning is required since the waiting time distribution tails of both VAD- and non-VAD-traffic match very closely. We note that for capacity allocation, the general interest lies in the tail distribution rather than the entire distribution. As a result, we consider non-VAD-traffic in the analysis without loss of generality; however, the results apply also to networks with VAD-traffic. In this chapter, we assume periodic arrivals of voice traffic generated by the same coders with equal rates. Each voice source generates a packet of fixed size M in the same periodicity of T seconds, producing traffic at rate $r = M/T$. The arrival times of packets belonging to the different K voice connections are assumed mutually independent and uniformly distributed over the time interval $[0, T)$.

It is worth emphasizing that we are interested in evaluating the minimum link capacity or capacity share that is required for voice traffic only. Doing so, we can check whether the available capacity is sufficient to serve this type of traffic according to its QoS constraints. This method solely is not enough for computing the total link capacity needed to serve a set of traffic classes (each with different QoS requirements). Currently, it is still not possible to devise a single method for IP networks that can evaluate the total capacity value required by all available traffic classes having different properties and quality constraints (e.g. data, voice, video, ...). To do so, other methods should be additionally considered to evaluate the capacity required per each class. The capacity values are then properly combined to obtain the total link capacity, which is capable of serving all supported classes with the desired quality. One direct approach of combining the individual capacity requirements is simply to sum them up [NQBM99] [RBF02]. Such approach might, however, lead to underutilized links since no multiplexing gain is considered.

Subsequently, in regards to the service process, the voice buffer is assumed to be served according to a first-in-first-out (FIFO) queue. C_{voice} represents the share of the link capacity responsible for serving voice traffic from its associated queue. Our task at this point is to evaluate C_{voice} so that all voice traffic is granted an acceptable quality at the buffer level with $\widehat{\mathcal{D}}_{\text{buf}}$ as the relevant QoS parameter that indicates the delay threshold at this level.

In the following section, we investigate different capacity assignment (CA) strategies for evaluating C_{voice} at the buffer level and at the node level. The latter detail level is also investigated at this point to account for the effect of the employed scheduling policy. Assuming that a maximum of K VoIP connections are forwarded to one output link as depicted in the buffer model in Figure 3.2, we need to find the minimum C_{voice}. Assuming that all VoIP sources implement the same encoders, K packets (belonging to the K active voice connections) will arrive within any time interval T. During network operation, a call admission control scheme can assure that not more than K connections are allowed on the link. Otherwise, the required QoS cannot be provided.

3.3 Capacity Assignment Strategies

As a network planner, the ultimate goal is to provide premium quality level for the supported services. Since we are concerned with capacity assignment for interactive voice, we need to evaluate the necessary link capacity share that strictly guarantees premium quality for this type of service. Granting hard guarantees means that *all* packets of *all* active K connections are definitely served within a delay threshold of $\widehat{\mathcal{D}}_{\text{buf}}$, irrespective of any factor. With "irrespective", we mean "even in the worst-case scenario for the given factor". The most relevant factor in our case is the arrival pattern of packets to the queue. The worst-case scenario is then defined as the case when all K packets of the K connections arrive at exactly the same time instant. It then has to be assured that the packet that happens to be put into the buffer last and that has to wait longest is still sent out within time $\widehat{\mathcal{D}}_{\text{buf}}$. Thus, $(K \cdot M)$ bytes have to be sent within $\widehat{\mathcal{D}}_{\text{buf}}$ requiring a rate of $(K \cdot M)/\widehat{\mathcal{D}}_{\text{buf}}$. This capacity assignment approach is based on worst-case considerations and thus referred to as worst-case capacity assignment or hard-guarantees capacity assignment. Figure 3.3 describes the worst-case scenario where the last packet is marked in dark. Deterministic upper bounds on queuing delays in packet-switched networks are derived in [PG93] [PG94] and they do also correspond to this worst-case scenario.

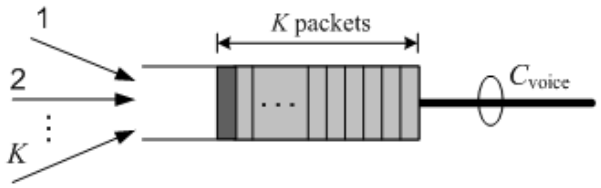


Figure 3.3: Worst-case scenario.

Dealing with an extreme worst-case capacity assignment, we wonder what the other extreme of the scale would be, namely best-case capacity assignment. Best-case CA approach corresponds to a scenario, in which every packet arrives at the moment when the previous packet has just been sent out (refer to Figure 3.4). Based on this scenario, the required capacity is at least $M/\widehat{\mathcal{D}}_{\text{buf}}$. However, we should be aware that the minimum possible capacity requirement should be at least the total bit rate of the K connections so as to assure a stable system.

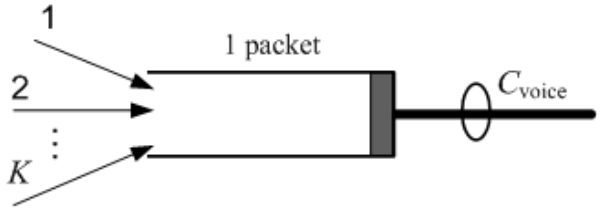


Figure 3.4: Best-case scenario.

In Figure 3.5, the range of capacity requirements for voice between the two extreme scenarios is illustrated for the following system parameters: $M = 200$ bytes, $T = 20$ ms, and $\widehat{\mathcal{D}}_{\text{buf}} = 5$ ms. For each of these extreme expectations, we plot the necessary C_{voice} so that the desired delay threshold is met. As the number of active voice connections K increases, the range gets broader and the difference of capacity requirements between the two extremes increases linearly with K. Using worst-case capacity assignment approach, capacity requirements are much higher than the total bit rate of all active connections causing very low link utilization. For example, for $K = 10$, only 25% of the link capacity share assigned to voice traffic is utilized ($\rho = (10 \cdot 80)/3200 = 0.25 = 25\%$). On the contrary, best-case CA requires a capacity slightly higher than the connections bit rate until $K = 4$ after which the best-case CA curve coincides with the bit rate curve achieving 100% link utilization.

While best-case capacity assignment certainly does not achieve satisfactory QoS in almost all cases (too many active connections would exceed their delay budget), worst-case capacity assignment is considered a completely pessimistic assumption [MWKB99]. Here rises the question, how necessary it is to dimension according to the worst-case capacity assignment approach. We can answer this question by computing $\mathrm{P}_{\text{worst}}$ that denotes the occurrence frequency or probability of the worst-case scenario on which the worst-case CA approach is based. We assume a slotted time interval T equivalent to the voice coder periodicity where each slot is

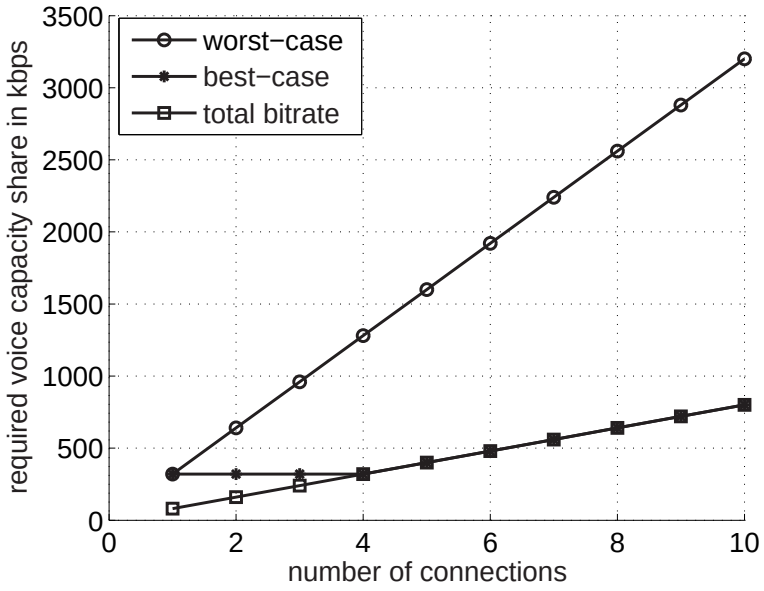


Figure 3.5: Worst-case vs. best-case capacity assignment at the buffer level.

equal to the service time T_s of one packet. The time interval T then contains N slots where $N = \lfloor T/T_\mathrm{s} \rfloor$. Packet arrivals are assumed to occur at the beginning of a slot and to be mutually independent and uniformly distributed over all available slots in T. Thus, the probability that a packet arrives at a given slot i, $i = 1, 2, \ldots, N$, evaluates to $p_i = 1/N$. Having equi-periodic sources, only one packet per active connection will appear every T interval. Therefore, the probability of the worst-case scenario is equivalent to the probability that all K packets of the K connections arrive in the same timeslot, for any timeslot, and it evaluates to

$$\mathrm{P}_\mathrm{worst} = \binom{N}{1} \cdot p_i^K = \frac{1}{N^{K-1}}. \tag{3.1}$$

If the link utilization is 100%, i.e. the number of connections K equals the number of timeslots N in T, $\mathrm{P}_\mathrm{worst}$ becomes

$$\mathrm{P}_\mathrm{worst} = \frac{1}{K^{K-1}}. \tag{3.2}$$

Setting $K = 10$ connections leads to $\mathrm{P}_\mathrm{worst} = 10^{-9}$. Therefore, we would be paying 300% additional capacity (see Figure 3.5) just to account for a case that occurs once every 10^9 times. For comparison purposes, the probability of the best-case scenario, P_best, in 100% link utilization is computed as well to be given by

$$\mathrm{P}_\mathrm{best} = \frac{K!}{K^K}. \tag{3.3}$$

Obviously, $\mathrm{P}_{\mathrm{best}}$ evaluates extremely higher than $\mathrm{P}_{\mathrm{worst}}$ especially for high values of K. Furthermore, for lower link utilization, $\mathrm{P}_{\mathrm{best}}$ is even higher. Setting $K = 10$ leads to $\mathrm{P}_{\mathrm{best}} = 3.6 \cdot 10^{-4}$. However, if dimensioning is done according to the best-case scenario, the desired QoS would not be achieved in $(1 - 3.6 \cdot 10^{-4})$ of all cases. It is clear that dimensioning according to either extreme is not realistic. While one approach wastes a lot of capacity, the other one leads to service degradation for most voice connections in most cases. A practical and satisfactory capacity assignment strategy should lie somewhere in between.

Table 3.1 shows the formulas used in computing the required capacity share for voice for various scheduling schemes. In the best-case scenario, only one voice packet is present in the queue and it is served immediately. Thus, C_{voice} should be high enough to serve this voice packet within the given delay threshold regardless of the scheduling scheme employed. In the worst-case scenario however, the employed scheduling scheme has an impact on the required capacity. For the buffer level and preemptive PQ, voice packets are served immediately as soon as they arrive to their queue assuming that voice traffic class is assigned the highest priority among the other classes. As a result C_{voice} has to be high enough so that all K packets with an overall size of $K \cdot M$ bytes are served within the given delay threshold. However, for non-preemptive PQ in the worst-case scenario, the K voice packets arrive at the empty queue at the instant when an MTU-size packet of another traffic class has just been put in service. Since the employed scheduling scheme is non-preemptive in nature, it waits until the packet at hand is fully transmitted. As a result, all K packets with an overall size of $K \cdot M$ bytes in addition to the MTU-size packet have to be sent out within $\widehat{\mathcal{D}}_{\mathrm{link}}$ time. Finally, for CB-WFQ, the pessimistic scenario is when the K voice packets arrive at the empty queue at the instant when an MTU-size packet of another traffic class has just been put in service. Therefore, we are left out with only $\widehat{\mathcal{D}}_{\mathrm{link}} - M_{\mathrm{MTU}}/C$ to send out all the K voice packets.

Table 3.1: Required C_{voice} for best-case and worst-case scenarios. We note that C_{voice} should be at least the total mean bit rate of all connections. In case C_{voice} is computed to be less than $K \cdot r$ then it is set to $K \cdot r$.

		Best case	**Worst case**
Buffer level		$\dfrac{M}{\widehat{\mathcal{D}}_{\mathrm{buf}}}$	$\dfrac{K \cdot M}{\widehat{\mathcal{D}}_{\mathrm{buf}}}$
Node level	Preemptive PQ	$\dfrac{M}{\widehat{\mathcal{D}}_{\mathrm{link}}}$	$\dfrac{K \cdot M}{\widehat{\mathcal{D}}_{\mathrm{link}}}$
	Non-preemptive PQ	$\dfrac{M}{\widehat{\mathcal{D}}_{\mathrm{link}}}$	$\dfrac{K \cdot M + M_{\mathrm{MTU}}}{\widehat{\mathcal{D}}_{\mathrm{link}}}$
	CB-WFQ	$\dfrac{M}{\widehat{\mathcal{D}}_{\mathrm{link}}}$	$\min\left(C, \dfrac{K \cdot M}{\widehat{\mathcal{D}}_{\mathrm{link}} - \frac{M_{\mathrm{MTU}}}{C}}\right)$

Instead of deterministic delay bounds, statistical delay values are considered. Taking into account that the K packets usually arrive in some way distributed over the time period T, a tradeoff capacity value for voice traffic seems to be sufficient. This way, softer QoS guarantees are acquired and the corresponding capacity assignment approach accounts for some kind of a tradeoff-case scenario, which is neither a worst-case nor a best-case scenario. However, accounting for a tradeoff-case scenario, one has to be aware that in some cases, not all of the

packets can be served within the desired delay budget. At this point, we need to investigate how frequent these cases occur, in which the delay budget is violated, and eventually provide a clear formulation for this new "tradeoff" capacity assignment approach.

For the above stated purpose, we define a $(K-i)$-packet worst-case scenario, where during a period T, the maximum number of packets in the buffer is $K-i$, with $i = 0, 1, \ldots, K-1$. Figure 3.6 presents the $(K-i)$-packet worst-case scenario. We need to assure that the packet queued last in the buffer (marked in dark in the figure) has to be transmitted within the allowed delay budget. Based on the definition of the $(K-i)$-packet worst-case scenario, the worst-case scenario corresponds to $i = 0$ and the best-case scenario corresponds to $i = K-1$. In the worst-case scenario, the last packet queued in the buffer waits until $K-1$ packets in front of it are served before its own service process starts. In consequence, it experiences a total delay of K service times and thus could be referred to as K-packet worst-case scenario. In the best-case scenario, each packet is served right away; so it experiences a delay of one service time overall and could be referred to as 1-packet worst-case scenario. Figure 3.7 plots the needed C_{voice} in terms of the number of active connections for all $(K-i)$-packet worst-case scenarios.

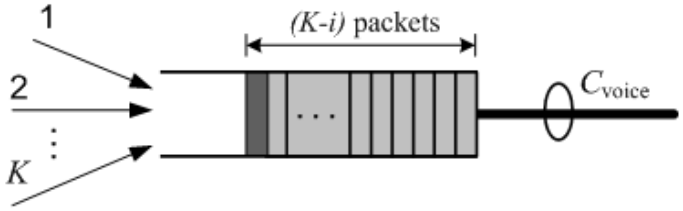


Figure 3.6: $(K-i)$-packet worst-case scenario.

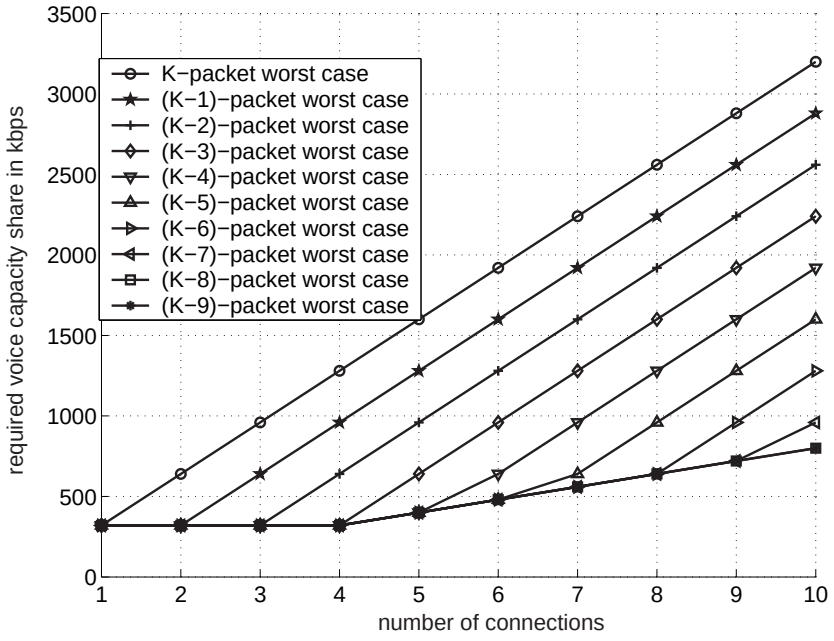


Figure 3.7: Required C_{voice} for $(K-i)$-packet worst case at the buffer level.

If dimensioning is carried out according to the $(K-2)$-packet worst-case scenario, 20% of the required capacity can be saved as compared to worst-case capacity assignment. This leads to service degradation only in $(K-0)$- and $(K-1)$-packet worst cases, which arise with a very low probability that amounts to 10^{-8} with 100% link utilization. We note that the probability of $(K-1)$-packet worst-case scenario is given by

$$\frac{\binom{K}{2} \cdot 2!}{K^K}$$

in a full load situation.

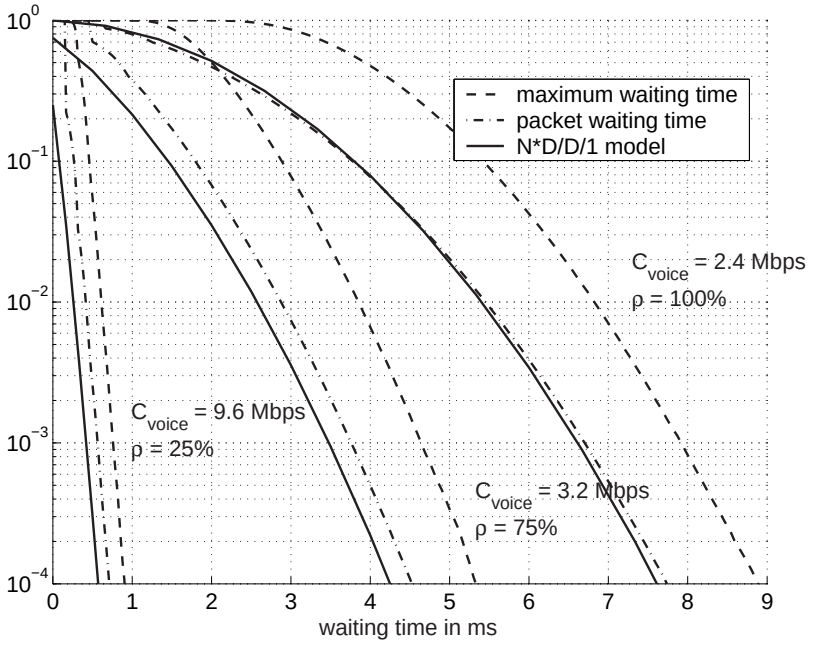
Using this $(K-i)$-packet worst-case concept, we can guarantee that in most cases (up to a certain probability) no packet has to wait more than the predefined delay threshold. Only in a few "unfortunate" cases, the threshold is exceeded.

3.3.1 Investigation of Waiting Time Models

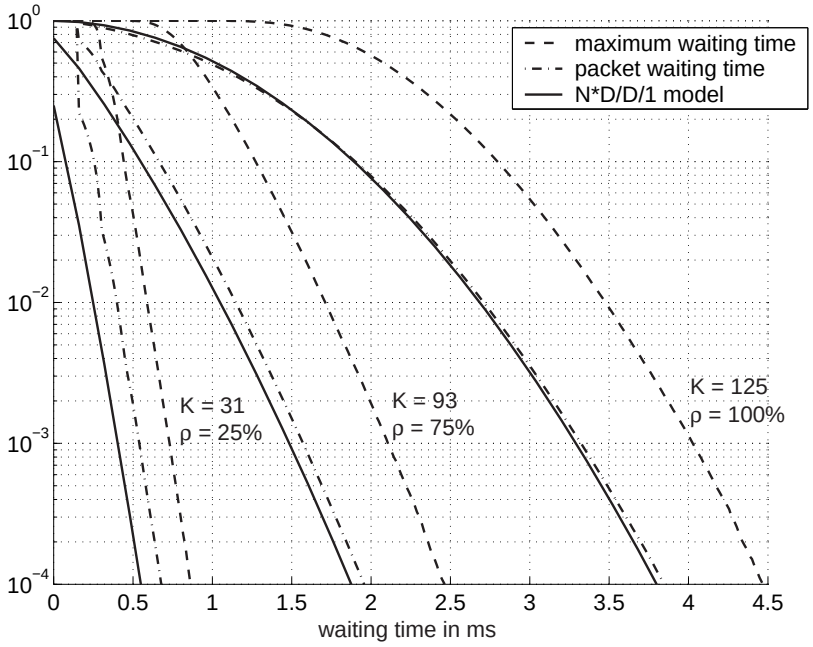
In this section, the probabilities of the above-mentioned "unfortunate" cases are evaluated by means of simulations. To do so, we generate arrival scenarios of K VoIP connections within a time interval of length T, which is equal to the coder period. The setup time of each connection is uniformly distributed over $[0, T)$. For each generated scenario, the packet waiting time and the maximum packet waiting time are computed, which can be converted into the number and maximum number of packets in the buffer.

The results are compared with an $N*D/D/1$ queuing system, which models a number of equi-periodic sources served at a constant rate by one server [RMV96]. This model is applicable to constant bit rate (CBR) voice traffic of equal coder rate multiplexed on one link [MWKB99] [BPR01]. While voice flows lose their strict periodic behavior in multiservice networks, it is preserved in networks with single voice service. As we investigate the waiting time occurring in the voice queue at the buffer level, it is apparent that modeling voice flows as $N*D/D/1$ is an appropriate approximation since the effect of co-existing traffic is ignored at this detail level.

In Figure 3.8a, we plot the complementary cumulative distribution function (CCDF) of the maximum waiting time encountered for 30 active connections in comparison with the packet waiting time and the $N*D/D/1$ model for various values of link utilization ρ (i.e. different link capacities). We observe, as expected, that the maximum waiting time curve is an upper bound for the packet waiting time which is in turn an upper bound for the $N*D/D/1$ model. For a fully utilized link, the CCDF of the packet waiting time matches very closely the $N*D/D/1$ model. This is clear, knowing that the packet waiting time records the delay of each packet, i.e, only when the queue is busy; whereas, the $N*D/D/1$ model computes the waiting time over all times even when the queue is empty. By having a fully utilized link, the queue is always occupied and, thus, both CCDFs match. In case the $N*D/D/1$ model is used for capacity assignment, it would suggest that a link speed of 3.2 Mbps (equivalent to 75% utilization) would be sufficient to keep the delay below 4.2 ms for a probability of $(1-10^{-4})$. However, the CCDF of the maximum waiting time at 4.2 ms is $4 \cdot 10^{-3}$. This means that in 4 out of 1000 cases, not all of the active VoIP connections experience the desired QoS.



(a)



(b)

Figure 3.8: CCDF of various waiting time models: $r = 80$ kbps, $\rho = 25\%$, 75%, and 100% (a) K is fixed equal to 30 (b) C_{voice} is fixed equal to 10 Mbps.

Figure 3.8b shows the different waiting time models for different link utilizations with fixed $C_{\text{voice}} = 10$ Mbps (this corresponds to different numbers of active connections). On a 10 Mbps link, 125 active connections could be multiplexed at one time (100% utilization) and a delay limit of 4.5 ms is guaranteed up to $(1 - 10^{-4})$ of the cases. Using the worst-case dimensioning approach with hard QoS guarantees and a delay threshold of 4.5 ms, 10 Mbps capacity would be required for only 28 active connections leading to an effective utilization of 22.4%. From a different perspective, the 125 active connections would require around 44 Mbps to have a hard guaranteed delay of 4.5 ms. This is due to the fact that worst-case capacity assignment corresponds to a 125-packet worst-case scenario. Whereas, the maximum delay model shows that a maximum of 28 packets are queued at one time for $(1 - 10^{-4})$ of the cases and it corresponds to 28-packet worst-case scenario. As a result, dimensioning VoIP networks based on the maximum queuing delay model provides "almost" guaranteed QoS for rather low costs.

At the node level, various traffic classes are available and thus it is important to assess the impact other classes will have on the voice class performance. We assume highly congested network conditions where queues of the available traffic classes excluding voice are constantly being filled up with MTU-size packets. Figure 3.9 presents the CCDF of the maximum queuing delay of voice packets awaiting service from a link of capacity C. Results are shown for each of non-preemptive PQ and CB-WFQ. It is observed that the performance using non-preemptive PQ is an upper bound to that of CB-WFQ when voice is served at the same rate by both scheduling schemes. This is simply due to the fact that in non-preemptive PQ, any packet of the other traffic classes that might have been in transmission when a voice packet arrives is served with the total link speed that is 4 Mbps. However, in CB-WFQ, any packet of the other classes that might have been in transmission when a voice packet is scheduled for service is served with the total link speed that is 20 Mbps and, thus, faster. However, with the same link capacity of 20 Mbps, PQ outperforms CB-WFQ even when 50% share is allocated for voice traffic.

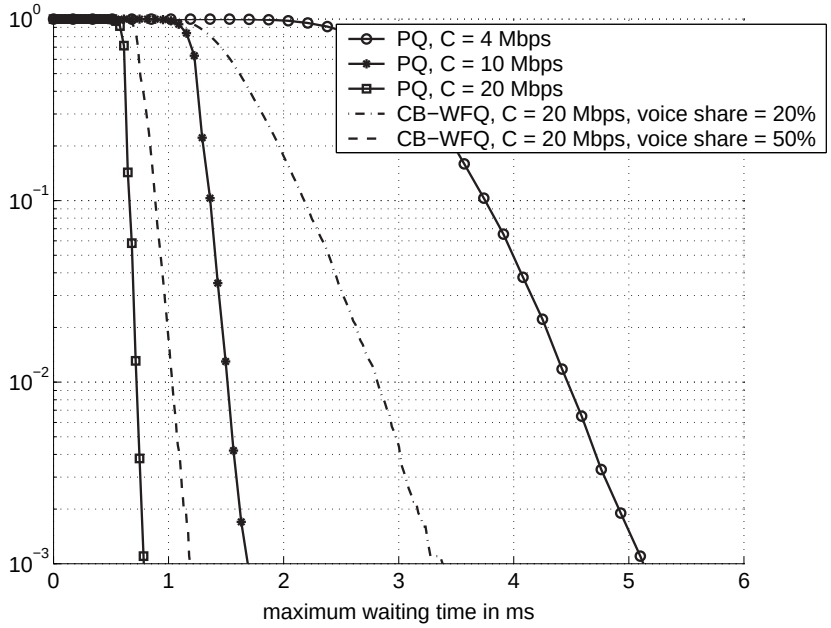


Figure 3.9: CCDF of maximum waiting time for various link capacities and scheduling schemes (at the node level): $r = 80$ kbps and $K = 10$ active connections.

3.4 A Novel Capacity Assignment Strategy

With the insights gained in the analysis of the previous sections, we propose a new capacity assignment method for voice traffic. As observed in Section 3.3.1, to properly allocate capacity for VoIP traffic, it is important to study the waiting time introduced in IP networks that should be restricted to a defined bound if high performance is desired. The allocated capacity is fully utilized to serve voice packets within the delay bound at times of pessimistic arrival scenarios which cause maximum waiting times. Therefore, if we know the maximum waiting time then we are able to determine the needed capacity.

The new method is established from the following concept: if we are aware of the packets that experience the maximum delay among all packets, we can practically protect all other packets from extra delay, and, thus, from quality degradation. This notion forms the basis of our proposed capacity allocation method. Slightly softened guarantees that allow the maximum waiting time among all packets to be exceeded very rarely lead to significant reduction in the capacity requirements. The proposed method offers such guarantees and it is referred to as the Almost Guaranteed QoS method. The method can be summarized by the following steps.

1. Calculate the *maximum* waiting time distribution of voice traffic aggregate at the voice queue of a network node.

2. Define a delay threshold $\widehat{\mathcal{D}}_{\mathrm{buf}}$ of the waiting time at the specific queue.

3. Define an outage probability P_{out} that represents the frequency of arrival patterns, in which the maximum waiting time is allowed to exceed $\widehat{\mathcal{D}}_{\mathrm{buf}}$ causing quality degradation.

4. Compute the link capacity such that the voice packet with the maximum waiting time at the given queue undergoes a delay that exceeds $\widehat{\mathcal{D}}_{\mathrm{buf}}$ in at most the defined outage probability P_{out}.

Though presented at only the buffer level as it is the basic level, the method description can be extended for any of the levels of our analysis-synthesis approach in Figure 3.1. We note that the allocation of one capacity value for a number of voice connections makes this method applicable to DiffServ networks, which provide service to behavior aggregates [BBC98]. So far, evaluation of the maximum waiting time distribution is performed by means of simulations. In the sequel, we intend to provide a mathematical analysis of the method.

To study the Almost Guaranteed QoS method mathematically, we start with the simplified model of the buffer level so as to obtain a feasible mathematical solution. Afterwards, we attempt to generalize the model gradually to account for new items at upper levels and extend the results accordingly [SKD04].

3.4.1 Maximum Waiting Time Process

We consider a process A that represents the work generated by the K active voice connections whose phases are assumed mutually independent and uniformly distributed over the period T. Each connection represents a periodic source that generates $1/N$ units of work every time interval T (i.e. $N = T/T_{\mathrm{s}}$, where T_{s} is the service time of one voice packet). Voice connections

generate equal-size packets of size M. The service rate of the resulting queuing system at the buffer level is the capacity share allocated to voice traffic, which is denoted as C_{voice}.

The workload $L(t)$ at time t in the system whose arrival process is A and service rate is C_{voice} is illustrated in Figure 3.10 and is defined by

$$L(t) = \frac{1}{M} \cdot \sup_{0 \le s \le t} \{A(t) - A(s) - (t-s) \cdot C_{\text{voice}}\}. \tag{3.4}$$

The queue length $U(t)$ is the number of packets in the queue and it is the integer part of $L(t)$. $L(t)$ is equivalent to the virtual waiting time process. Thus, the waiting time process $W(t)$ of this system has a similar distribution as $L(t)$ except for a scaling factor knowing that $W(t) = L(t) \cdot T_{\text{s}}$.

The maximum waiting time in the system is then given by

$$\widehat{W} = \max_{t \ge 0} W(t). \tag{3.5}$$

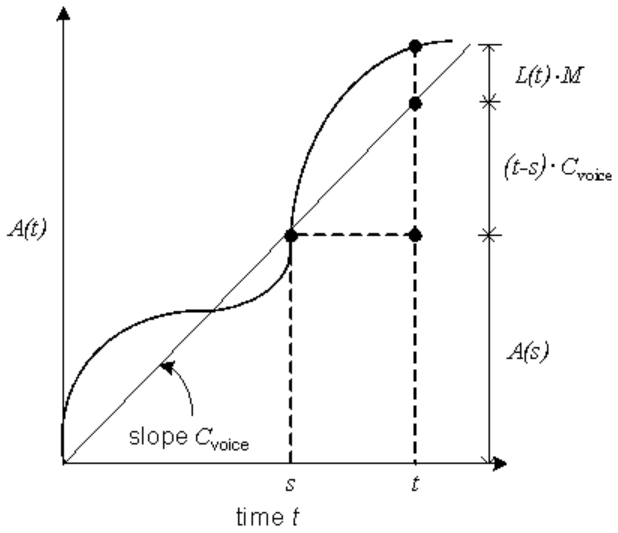


Figure 3.10: Computation of the workload in the system.

3.4.2 Analysis at the Buffer Level

To start with, we consider the buffer model to which K active voice connections are directed. By doing so, we study certain aspects of this traffic in isolation to various factors. Among these factors, there exists the effect introduced by other interfering traffic competing for the available capacity and the effect introduced by the existence of multiple hops from source to destination. At the buffer level, the resulting queuing system is reduced to a queue with periodic arrivals and constant service times. The resulting scenario can be modeled as an $N * D/D/1$ queuing system, which models a superposition of N independent equi-periodic sources served

in constant service time. The output process is considered as an equal-slotted transmission channel that can transmit exactly one packet per timeslot (i.e. the length of a timeslot is equal to the service time per packet $T_\mathrm{s} = M/C_\mathrm{voice}$).

To compute the necessary capacity for VoIP service using the proposed method, the maximum waiting time model of the traffic is required as a first step. The virtual waiting time distribution (unfinished work) of the $N * D/D/1$ model had been extensively studied in the literature [Eck79] [HBH93] [RV91], while the maximum waiting time distribution is briefly handled and is not widely applied for practical purposes. The unfinished work, however, if used for capacity allocation, results in underestimated capacity values since it accounts for zero unfinished work i.e. when the queue is empty. The maximum waiting time, on the other hand, is shown by simulations in Section 3.3.1 to be more appropriate for capacity allocation purposes when QoS guarantees are required as in VoIP.

To obtain the maximum waiting time distribution of the $N*D/D/1$ model, we return to methods given by Ott and Shantikumar in [OS91] for a discrete time model and generalized by Hajek in [Haj94] for the continuous case. In our model, K active voice connections are routed over the channel. Each connection sends a packet once every N timeslots (i.e. $N = \lfloor T/T_\mathrm{s} \rfloor$). Eventually, the channel can handle at most N connections simultaneously, i.e. $K \leq N$. In the following, the maximum waiting time distribution is computed in a discrete-time model and then generalized in a continuous-time model.

3.4.2.1 Maximum Waiting Time Distribution in Discrete-Time Model

In the discrete time model, the distribution of $\widehat{W}$ is obtained from the distribution of maximal queue length (i.e. maximum number of packets in the queue). Let Y_t be the number of packets entering the queue during timeslot t. The queue length behavior studied in [OS91] is based on the fact that the stochastic process $(Y_t)_{t=-\infty}^{+\infty}$ is N-periodic and a strongly interchangeable process[1]. This corresponds to our model since K active voice connections are assumed to be independent and each connection sends a packet during one of the timeslots $t = 1, 2, \ldots, N$ where each of the N timeslots is equally likely. Then,

$$Y_t = Y_{t+kN} \quad \forall k \in \mathbb{Z}, \tag{3.6}$$

$$\sum_{t=1}^{N} Y_t = K \quad 0 \leq K \leq N, \tag{3.7}$$

$$\mathrm{P}\left\{Y_1 = y_1, \cdots, Y_N = y_N\right\} = \binom{K}{y_1, y_2, \ldots, y_N} \left(\frac{1}{N}\right)^K, \tag{3.8}$$

where

$$\binom{K}{y_1, y_2, \ldots, y_N} = \frac{K!}{y_1! \cdot y_2! \ldots y_N!}.$$

[1]The discrete-time stochastic process $(Z_t)_{t=-\infty}^{+\infty}$ is called N-*periodic* if $Z_t = Z_{t+kN}$ for all integers k, and is called *strongly interchangeable* if $(Z_{t+1}, \cdots, Z_{t+N})$ are interchangeable random variables, and for every subset $\mathcal{S}$ of $\{1, 2, ..., N\}$, given $\sum_{k\in\mathcal{S}} Z_{t+k} = K$, $(Z_{t+k})_{k\in\mathcal{S}}$ and $(Z_{t+k})_{k\in\bar{\mathcal{S}}}$ are conditionally independent ($\bar{\mathcal{S}}$ is the complementary set of $\mathcal{S}$).

Clearly, the process $(Y_t)_{t=-\infty}^{+\infty}$ in (3.6) – (3.8) is N-periodic and strongly interchangeable. The random variables $Y_1, \ldots, Y_N$ can be thought of as independent and identically distributed (i.i.d) Poisson random variables with the condition that $\sum_{t=1}^{N} Y_t = K$.

Let U_t be the number of packets in the queue at the end of timeslot t for all t integers. The stationary process $(U_t)_{t=0}^{\infty}$ is defined by

$$\begin{array}{lcll} U_0 & = & 0, & \\ U_{t+1} & = & (U_t - 1)_+ + Y_{t+1} & \text{if } t \geq 1, \\ U_t & = & U_{t+kN} & \forall k \in \mathbb{Z}, \end{array} \tag{3.9}$$

where $(x)_+ = \max(x, 0)$.

If $K = N$, the additional constraint

$$\min_{1 \leq k \leq N} U_{t+k} = 1$$

uniquely defines the stationary distribution of maximal queue length $\widehat{U}$ at departure instant, where $\widehat{U} = \max_{1 \leq k \leq N} U_{t+k}$. The probability that the maximum number of packets in the queue $\widehat{U}$ exceeds a constant integer n, where $n = 0, 1, \ldots, K-1$, is given by [OS91] as

$$\mathrm{P}\left\{\widehat{U} > n\right\} = \begin{cases} 1 - \left(\dfrac{K!}{N^{K-1}(N-K)} e^N r_N^{(N-K)}(n+1) \right) & \text{if } 0 \leq K \leq N-1, \\ 1 - \left(\dfrac{N! e^N}{N^{N-1}} \displaystyle\sum_{j=1}^{N} \frac{1}{j} r_N^{*(j)}(n+1) \right) & \text{if } K = N, \end{cases} \tag{3.10}$$

where $r_k(n)$, for k integer, is the probability that during a typical busy period of an $M/D/1$ queue (arrival rate = $\lambda = \frac{N}{1\,\mathrm{sec}}$, service time = $\frac{1\,\mathrm{sec}}{N}$) starting at time $t = 0$, exactly k customers arrive and are served (i.e. busy period has finite length equal to k service times) and the virtual waiting time (in unit of service times) remains in $[0, n)$ throughout the busy period. The term $r_k^*(n)$ denotes the probability that the $M/D/1$ system returns to the situation where the virtual waiting time at a customer departure epoch equals 1 for the first time at the end of timeslot k, and that in the time interval $[0, k \cdot T_\mathrm{s})$ the virtual waiting time remains in $[0, n)$. The factor $r_k^{(j)}(n)$ and $r_k^{*(j)}(n)$ denote the j-fold convolution of $r_k(n)$ and $r_k^*(n)$ respectively.

The probabilities $r_k(n)$ and $r_k^{(j)}(n)$ can be computed in the following way.

Let $\Pi(n)$ be the $n \times n$ matrix,

$$\Pi(n) = \begin{bmatrix} \pi_1 & \pi_2 & \cdots & \pi_n \\ \pi_0 & \pi_1 & \cdots & \pi_{n-1} \\ 0 & \pi_0 & \cdots & \pi_{n-2} \\ \vdots & \vdots & \vdots & \vdots \\ \cdots & 0 & \pi_0 & \pi_1 \end{bmatrix},$$

where $\pi_i = e^{-1}/i!$, $i = 0, 1, \ldots, n$, and $\pi_1^{[k]}(n)$ is the top leftmost element of the k-th power of matrix $\Pi(n)$, $(\Pi(n))^k$, for integer $k \geq 0$. Hence, $r_k(n)$ can be calculated as

$$r_k(n) = \begin{cases} 0 & \text{if } k \leq 0, \\ e^{-1} \pi_1^{[k-1]}(n-1) & \forall k \in \mathbb{Z}, \end{cases} \tag{3.11}$$

and the j-fold convolution $r_k^{(j)}(n)$, for any integer $j \geq 0$, can be iteratively calculated as follows.

$$\begin{aligned} r_k^{(0)}(n) &= \delta(k,0), \\ r_k^{(j)}(n) &= \sum_{i=1}^{k} r_i(n) r_{k-i}^{(j-1)}(n) \quad \text{if } j > 0, \end{aligned} \tag{3.12}$$

where $\delta(x,y) = 1$ for $x = y$ and $\delta(x,y) = 0$ otherwise.

The term $\sum_{j=1}^{N} \frac{1}{j} r_N^{*(j)}(n+1)$ in (3.10) for $K = N$ is computed using the algorithm in [OS91]. Let

$$\begin{aligned} b_0(n) &= 0, \\ b_k(n) &= \sum_{j=1}^{k} \frac{1}{j} r_k^{*(j)}(n), \end{aligned} \tag{3.13}$$

and $\sum_{j=1}^{N} \frac{1}{j} r_N^{*(j)}(n) = b_N(n)$ is iteratively calculated using

$$\begin{aligned} b_1(n) &= \pi_1 = e^{-1}, \\ b_k(n) &= \pi_1^{[k]}(n-1) - \frac{1}{k} \sum_{i=1}^{k-1} i \cdot b_i(n) \pi_1^{[k-i]}(n-1) \quad \text{if } k = 2, ..., N. \end{aligned} \tag{3.14}$$

Now that all components of (3.10) are computable, we can form the distribution of $\widehat{U}$. Since the maximum waiting time occurs at the time when the maximum number of packets are present in the queue, we can relate the maximum number of packets $\widehat{U}$ in the queue viewed at packet departure instant to the maximum waiting time $\widehat{W}$ for constant service time as

$$\widehat{W} = \left(\widehat{U} + 1\right) \cdot T_{\mathrm{s}} \tag{3.15}$$

The addition of 1 to $\widehat{U}$ in (3.15) is required due to the fact that $\widehat{U}$ is associated with the maximal queue length at the end of any timeslot. Knowing that *one* packet departs at the end of a timeslot leaving behind a queue with maximum number of packets equals $\widehat{U}$, the maximum number of packets in the queue considered at any time within the given timeslot is equivalent to the number of packets in the queue before the departure instant, i.e. $(\widehat{U} + 1)$.

Using the relations in (3.10) and (3.15), we obtain the CCDF of $\widehat{W}$ in discrete time for $n = 0, 1, \ldots, K$, as

$$\begin{aligned} P\{\widehat{W} > n \cdot T_{\mathrm{s}}\} &= P\{(\widehat{U}+1) \cdot T_{\mathrm{s}} > n \cdot T_{\mathrm{s}}\} \\ &= P\{\widehat{U} > n-1\} \\ &= \begin{cases} 1 - \left(\dfrac{K!}{N^{K-1}(N-K)} e^N r_N^{(N-K)}(n) \right) & 0 \leq K \leq N-1, \\ 1 - \left(\dfrac{N! e^N}{N^{N-1}} \displaystyle\sum_{j=1}^{N} \frac{1}{j} r_N^{*(j)}(n) \right) & K = N. \end{cases} \end{aligned} \tag{3.16}$$

3.4.2.2 Maximum Waiting Time Distribution in Continuous-Time Model

The maximum waiting time distribution of the $N * D/D/1$ queue is generalized by Hajek in [Haj94] for the continuous-time model. The virtual waiting time process of an $N * D/D/1$ queue is stationary and periodic with period equals to T and has the same distribution as the virtual waiting time process in an $M/D/1$ queue with the condition that K customers arrive during the interval $[0, T)$ and the server is idle at time T. The probability that the maximum waiting time $\widehat{W}$ exceeds a constant say $\widehat{\mathcal{D}}_{\text{buf}}$, where $0 \leq \widehat{\mathcal{D}}_{\text{buf}} \leq K \cdot T_{\text{s}}$, is given by

$$P\{\widehat{W} > \widehat{\mathcal{D}}_{\text{buf}}\} = \begin{cases} 1 - \dfrac{\sum_{j=1}^{K} r_K^{(j)}(\widehat{\mathcal{D}}_{\text{buf}}) f\left(\lambda\left(1 - \frac{K}{N}\right), j\right)}{\left(1 - \frac{K}{N}\right) f(\lambda, K)} & 0 \leq K \leq N-1, \\ 1 - \dfrac{r_K(\widehat{\mathcal{D}}_{\text{buf}})\lambda}{f(\lambda, K)} & K = N, \end{cases} \tag{3.17}$$

where $r_K(\widehat{\mathcal{D}}_{\text{buf}})$ is the probability that during a typical busy period of an $M/D/1$ queue (arrival rate $= \lambda = \frac{N}{1\,\text{sec}}$, service time $= \frac{1\,\text{sec}}{N}$) starting at time $t = 0$, exactly K customers arrive and are served, and the virtual waiting time remains in $\left[0, \widehat{\mathcal{D}}_{\text{buf}}\right)$ throughout the busy period. The factor $r_K^{(j)}(\widehat{\mathcal{D}}_{\text{buf}})$ denotes the j-fold convolution of $r_K(\widehat{\mathcal{D}}_{\text{buf}})$ and $f(\mu, i) = e^{-\mu} \cdot \mu^i / i!$. To compute the probability $r_K(\widehat{\mathcal{D}}_{\text{buf}})$ and its j-fold convolution $r_K^{(j)}(\widehat{\mathcal{D}}_{\text{buf}})$, we use a similar way as described in the discrete time model using (3.11) and (3.12) respectively by approximating the delay bound $\widehat{\mathcal{D}}_{\text{buf}}$ to an n integer multiples of T_{s}, i.e. $\widehat{\mathcal{D}}_{\text{buf}} = \left\lfloor \widehat{\mathcal{D}}_{\text{buf}}/T_{\text{s}} \right\rfloor \cdot T_{\text{s}} + \varepsilon \approx n \cdot T_{\text{s}}$, where $n = \left\lfloor \widehat{\mathcal{D}}_{\text{buf}}/T_{\text{s}} \right\rfloor$ is an integer and $0 \leq \varepsilon \leq T_{\text{s}}$. As a result, $r_K(\widehat{\mathcal{D}}_{\text{buf}}) \approx r_K(n)$ in (3.11) and $r_K^{(j)}(\widehat{\mathcal{D}}_{\text{buf}}) \approx r_K^{(j)}(n)$ calculated from (3.12).

Now, the maximum waiting time distribution can be computed in the discrete and continuous-time models using (3.16) and (3.17), respectively. In Figure 3.11, we plot the computed $\widehat{W}$ distributions using the continuous and discrete time models and compare them to Monte-Carlo simulation results. In the simulation, the buffer model presented in Section 3.2.4 is implemented, where we generate T-periodic arrivals of K voice connections with uniformly distributed arrival time over $[0, T)$. For each new set of arrivals, we compute the maximum waiting time encountered among all packets. For $K = 75$ voice connections, the maximum waiting time exceeds 1.5 ms with a probability of 10^{-3} (refer to Figure 3.11). We observe that $\widehat{W}$ CCDF computations produce quite accurate results especially in the continuous-time model. Simulated and computed $\widehat{W}$ CCDF plots corresponding to the same system parameters almost overlap. If K voice connections are active, a delay threshold of $\widehat{\mathcal{D}}_{\text{buf}}$ is allowed, and an outage probability of P_{out} is tolerated, then we can iteratively solve for the required capacity C_{voice} using (3.17) by substituting $N = \left\lfloor \frac{T}{T_{\text{s}}} \right\rfloor = \left\lfloor \frac{T}{(M/C_{\text{voice}})} \right\rfloor$.

Example 3.1 *$K = 50$ active voice connections are multiplexed onto one outgoing link of a network node having voice capacity share of C_{voice}. Each voice connection generates $M =$*

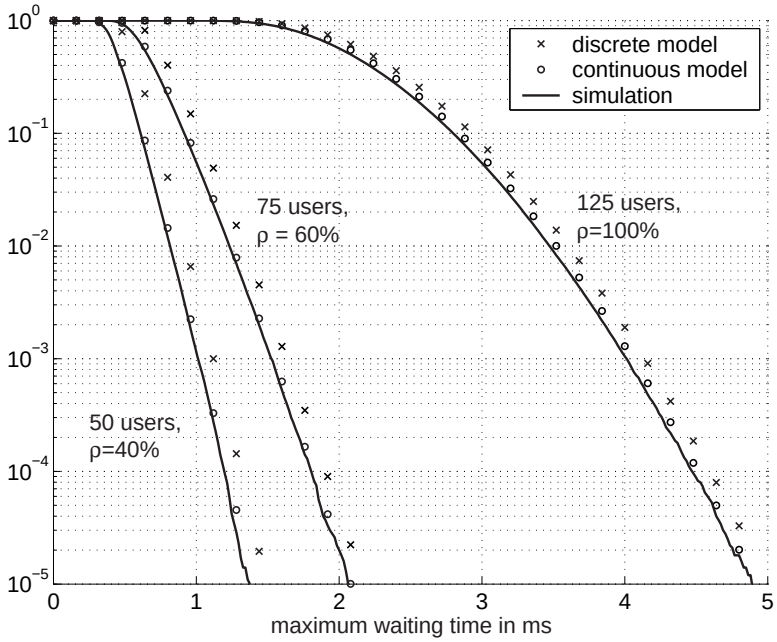


Figure 3.11: Calculation and simulation results of the CCDF of the maximum waiting time $\widehat{W}$: $C_{\text{voice}} = 10$ Mbps, $M = 200$ bytes, $T = 20$ ms, $K = [50\ 75\ 125]$ users corresponding to $\rho = [40\%\ 60\%\ 100\%]$ respectively.

200*-byte packet every* $T = 20$ *ms. Compute* C_{voice} *that is needed to ensure a high quality communication, where a delay threshold* $\widehat{\mathcal{D}}_{\text{buf}} = 5$ *ms is allowed to be exceeded with at most an outage probability* $P_{\text{out}} = 10^{-5}$.

Using the Almost Guaranteed QoS method, we calculate C_{voice} using (3.17) as follows

$$P\{\widehat{W} > 5\} = 1 - \frac{\displaystyle\sum_{j=1}^{50} r_{50}^{(j)}(5) f\left(\left\lfloor \frac{C_{\text{voice}}}{80} \right\rfloor \cdot \left(1 - \frac{50}{\left\lfloor \frac{C_{\text{voice}}}{80} \right\rfloor}\right), j\right)}{\left(1 - \frac{50}{\left\lfloor \frac{C_{\text{voice}}}{80} \right\rfloor}\right) f\left(\left\lfloor \frac{C_{\text{voice}}}{80} \right\rfloor, 50\right)} \leq 10^{-5}.$$

Solving numerically for C_{voice}, we get $C_{\text{voice}} = 4960$ kbps = 4.96 Mbps. On a 4.96 Mbps link, 50 active connections can be simultaneously multiplexed ($\rho = \frac{50 \cdot 80}{4960} \approx 81\%$) and a delay limit of 5 ms is guaranteed except for an outage probability of 10^{-5}. We note that the utilization is less than one which validates the usage of the first expression in (3.17). Using the worst-case capacity assignment approach (i.e. $P_{\text{out}} = 0$) with a delay threshold of 5 ms, a capacity of 16 Mbps is required for the same number of connections ($K = 50$) which leads to an effective utilization of 25%. Therefore, using the Almost Guaranteed QoS method based on the maximum waiting time model, the amount of required capacity can be reduced by 69%. ■

3.4.3 Analysis at the Node Level

In an IP network supporting several traffic classes, a scheduling policy is employed to manage the number of queues contained in the network node. In general, the existence of other queues/traffic classes sharing part of the available capacity negatively affects the performance of the voice service by introducing extra delay to voice packets that wait for their turn of service among other queues. As a pessimistic assumption, packets of other traffic classes are assumed MTU-sized and are constantly filling up their queues. In the following, we attempt to compute the distribution of $\widehat{W}$ at the node level for PQ and CB-WFQ scheduling schemes.

3.4.3.1 Priority Queuing

If PQ is the scheduling scheme used, voice traffic is assigned to a separate first-in-first-out (FIFO) queue with the highest priority. When the preemptive type of PQ is applied, voice traffic class is strongly protected against other traffic classes as if it is the only class available in the network. So to speak, the $\widehat{W}$ distribution at the node level maintains the same distribution obtained at the buffer level and thus the same results presented in Section 3.4.2 do also apply at the node level.

However, when the non-preemptive version of PQ is used, deterioration in the voice class performance is observed due to the interference caused by other traffic classes. The waiting time incurred on voice packets is affected by the lower priority traffic. This happens when a voice packet arrives to an empty voice queue at the time when a lower priority packet is in service; then, the voice packet has to wait an additional time before it receives its turn of service. This additional time is simply the time needed to complete the transmission of the lower priority packet observed at the arrival instant of the voice packet and it is referred to as *residual time* and denoted as w_{residue}. As a result, voice traffic periodicity (also observed in Chapter 2) is negatively influenced when a variable residual time caused by transmission of packets of other classes is added. In this section, we examine the resulting queuing model at the node level so as to extend results of the previous section to include residual transmission time of lower priority packets.

Figure 3.12 shows the CCDF of $\widehat{W}$ in non-preemptive PQ case and corresponding to a set of Monte-Carlo simulations with various MTU-size packets starting from 0 bytes up to 2500 bytes. In our simulations, we account for a residual service time of a lower priority packet, uniformly distributed over $[0, \frac{M_{\text{MTU}}}{C_{\text{voice}}}]$, at the instant when a voice packet just arrives to an empty voice queue. From Figure 3.12, it is obvious that the introduction of another traffic class in the network causes $\widehat{W}$ distribution of voice traffic to deviate from its initial distribution at the buffer level. It is realized that the deviation distance (calculated in percentage of one service time of the corresponding MTU, $T_{\text{MTU}} = \frac{M_{\text{MTU}}}{C_{\text{voice}}}$) does not exceed T_{MTU} for any value of M_{MTU} and it increases as the MTU size increases. Such observation is logical since the inclusion of MTU-size traffic in addition to voice traffic affects the maximum waiting time of voice traffic by at most T_{MTU}.

The maximum waiting time among all voice packets can be formulated as

$$\widehat{W} = \max(w_{\text{voice}} + w_{\text{residue}}), \tag{3.18}$$

where w_{voice} is the waiting time resulted from queuing behind other voice packets in the same queue, and w_{residue} is the residual transmission time of lower priority packets regardless to

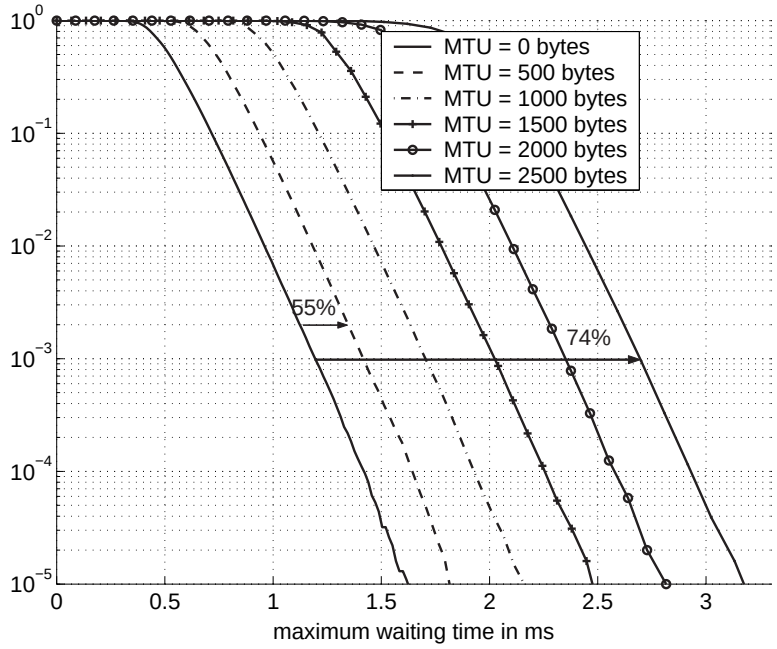


Figure 3.12: $\widehat{W}$ CCDF using non-preemptive PQ: $K = 60$ $M = 200$ byte, $T = 20$ ms and $C_{\text{voice}} = 10$ Mbps.

which class they belong to. The distribution of w_{residue} is typically assumed to be uniform in the interval $[0, T_{\text{MTU}}]$ [Kle75]. We have verified this fact by means of simulations and found out that it is more applicable to networks with low to medium load. When the network load increases further, the distribution of w_{residue} gets to be deviated from the uniform distribution. A lower bound on $\widehat{W}$ is achieved when no lower priority packets are being served when a voice packet arrives to an empty queue. This scenario yields similar results to those obtained at the buffer level, where the effect of all other traffic classes on voice transmission is ignored. Therefore,

$$\begin{aligned} \widehat{W} &= \max(w_{\text{voice}} + w_{\text{residue}}) \\ &\geq \max(w_{\text{voice}}) = \widehat{W}|_{\text{buffer}}, \end{aligned} \tag{3.19}$$

where $\widehat{W}|_{\text{buffer}}$ denotes the maximum waiting time at the buffer level, which is computed in Section 3.4.2.

On the other hand, an upper bound on $\widehat{W}$ is achieved when the specific packet that experiences the maximum waiting time at the buffer level arrives to the queue at exactly the same time when a full MTU-size packet has just been placed in service, that is

$$\begin{aligned} \widehat{W} &= \max(w_{\text{voice}} + w_{\text{residue}}) \\ &\leq \max(w_{\text{voice}}) + \max(w_{\text{residue}}) \\ &\leq \widehat{W}|_{\text{buffer}} + T_{\text{MTU}}, \end{aligned} \tag{3.20}$$

where w_{voice} and w_{residue} are assumed mutually independent. The distributions resulting from the upper and lower bounds are computed and plotted in Figure 3.13, where they are compared to Monte-Carlo simulation results. The CCDF of the maximum waiting time of the simulation

results are viewed to fall in between the two bounds. This observation verifies the computed lower and upper bounds in (3.19) and (3.20) respectively.

In the light of the above analysis, we conjecture that the actual $\widehat{W}$ can be well approximated by

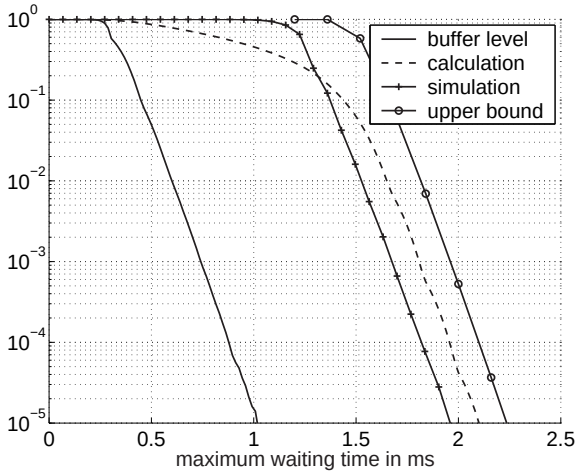
$$\widehat{W} \approx \max(w_{\text{voice}}) + w_{\text{residue}} = \widehat{W}|_{\text{buffer}} + w_{\text{residue}} \tag{3.21}$$

at quantile probabilities, since the influence of voice traffic with enough number of connections increases at low probabilities and becomes the dominant factor of the delay percentile as compared to the residual transmission time of only one MTU packet. So, whether (3.18) or (3.21) is used, very close $\widehat{W}$ distributions are resulted as illustrated in Figure 3.13 where the simulation curve refers to (3.18) and the calculation curve refers to (3.21). We note that the calculation of $\widehat{W}$ according to (3.21) assumes that $\widehat{W}|_{\text{buffer}}$ and w_{residue} are independent. The distribution of $\widehat{W}$ is obtained by convolution of $\widehat{W}|_{\text{buffer}}$ distribution and the uniform distribution of w_{residue},

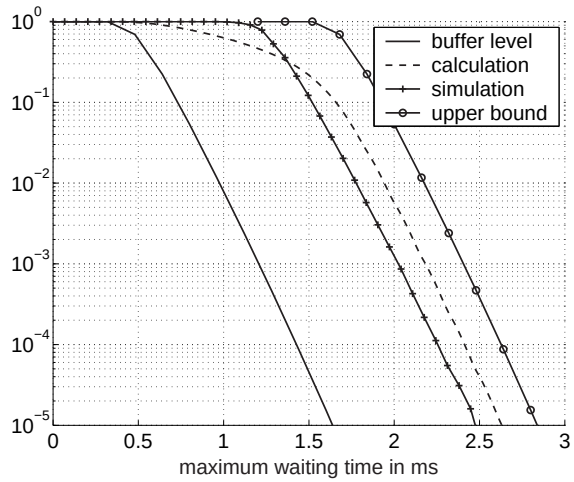
$$\begin{aligned} \mathrm{P}\{\widehat{W} > \widehat{\mathcal{D}}_{\text{link}}\} &\approx \mathrm{P}\{\widehat{W}|_{\text{buffer}} + w_{\text{residue}} > \widehat{\mathcal{D}}_{\text{link}}\} \\ &\approx 1 - \mathrm{P}\{\widehat{W}|_{\text{buffer}} + w_{\text{residue}} \leq \widehat{\mathcal{D}}_{\text{link}}\} \\ &\approx 1 - \int_0^{T_{\text{MTU}}} F_{\widehat{W}|_{\text{buffer}}}(\widehat{\mathcal{D}}_{\text{link}} - x) \cdot f_{w_{\text{residue}}}(x)dx, \end{aligned} \tag{3.22}$$

where $F_{\widehat{W}|_{\text{buffer}}}(x)$ denotes the cumulative distribution function (CDF) of $\widehat{W}|_{\text{buffer}}$ and $f_{w_{\text{residue}}}(x)$ is the probability density function (PDF) of w_{residue} which is uniform over $[0, T_{\text{MTU}})$. It is observed in Figure 3.13 that the calculation curve falls below the simulation curve at high probability values (i.e. underestimating the actual maximum waiting time). We note that the actual $\widehat{W}$ is the time when the maximum residual transmission time occurs, while the approximate $\widehat{W}$ (calculated in (3.21)) adds a uniformly distributed residual transmission time. The simulation and calculation curves cross at higher probability values when the influence of voice traffic increases. For higher voice loads, the two curves cross at higher probability values. This is due to the fact that voice traffic gets to have more effective influence on $\widehat{W}$. For example, in Figure 3.13a where voice load amounts to 25%, the simulation and the calculation curves cross at a probability of 0.2. When the voice load increases to 50% in Figure 3.13b and 80% in Figure 3.13c, the two curves cross at 0.3 and 0.8, respectively.

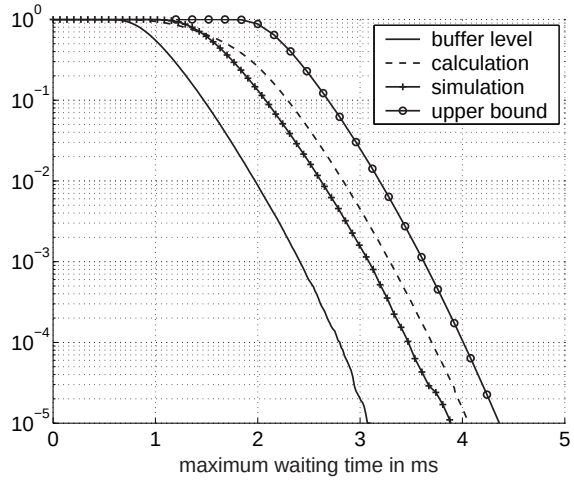
However, for capacity allocation purposes, the interest lies in the tail distribution rather than the whole distribution. Therefore, the CCDF of the approximated $\widehat{W}$ in (3.21) can serve as an appropriate mathematical model in calculating the required C_{voice} for a desirable quality. For example, using the approximated $\widehat{W}$ of (3.21) to compute C_{voice} if $K = 60$ connections, $\widehat{\mathcal{D}}_{\text{link}} = 2.4$ ms, and $P_{\text{out}} = 10^{-4}$, we obtain $C_{\text{voice}} = 10$ Mbps. It is shown by simulations (refer to Figure 3.13b) that 10 Mbps are already sufficient to guarantee a delay limit of 2.26 ms for $P_{\text{out}} = 10^{-4}$. Therefore, using the approximated $\widehat{W}$, we can provide a safe dimensioning for voice traffic. Yet, if we still use the upper bound in (3.20) and compare results to the worst-case capacity assignment approach, the latter requires $C_{\text{voice}} = \frac{(60 \cdot 200 + 1500) \cdot 8}{2.6 \cdot 10^{-3}} = 41.6$ Mbps to provide a deterministic delay bound of 2.6 ms while the former demands only 10 Mbps for the same delay threshold but with an outage probability of $P_{\text{out}} = 10^{-4}$.



(a) Low voice load: $\rho = 25\%$.



(b) Medium voice load: $\rho = 50\%$.



(c) High voice load: $\rho = 80\%$.

Figure 3.13: $\widehat{W}$ CCDF with non-preemptive PQ: $C_{\text{voice}} = 10$ Mbps, $M_{\text{MTU}} = 1500$ bytes, $M = 200$ bytes, $T = 20$ ms, and $r = 80$ kbps.

3.4.3.2 Class-Based Weighted Fair Queuing

In case CB-WFQ is employed, C_{voice} represents a service rate that is an explicit share Φ of the link capacity C, hence $C_{\text{voice}} = \Phi \cdot C$. Though assigned a fixed share of the link capacity, the existence of other traffic classes still affects the voice class. Knowing that CB-WFQ is work-conserving, this occurs when a voice packet arrives after the time when its associated queue is being scheduled for service but skipped to serve the next traffic class in turn since the voice queue is empty then. As the scheduling scheme is non-preemptive in nature, the arriving voice packet has to wait in the queue until the packet in service is completely transmitted, causing a variable residual time w_{residue}. A pessimistic scenario is that other traffic classes have MTU-size packets which are constantly filling up their queues. Thus, the residual transmission time of other traffic classes can be at most $T_{\text{MTU}} = \frac{M_{\text{MTU}}}{C}$.

Figure 3.14 shows the CCDF of $\widehat{W}$ corresponding to a set of Monte-Carlo simulations with varying MTU-size packets. It is observed that the simulated curves deviate from the original curve obtained at the buffer level. The deviation distances are indicated in the figure in percentages of T_{MTU} and it is noticed that none exceeds T_{MTU} for any value of M_{MTU}.

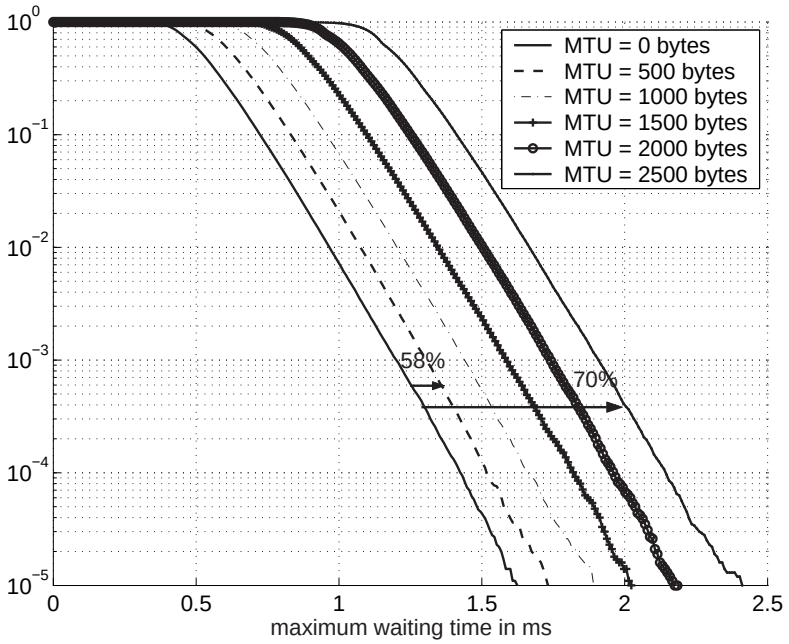
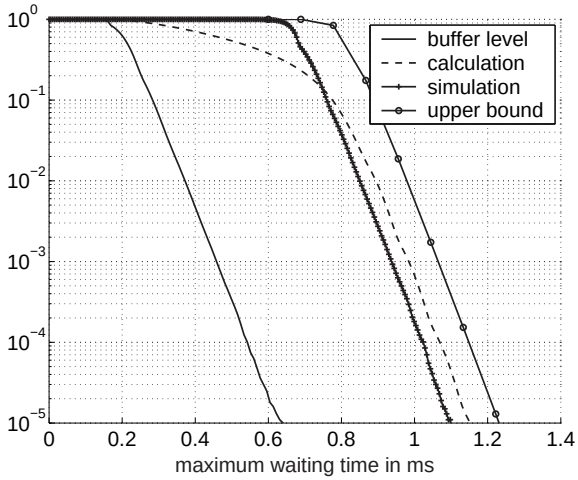
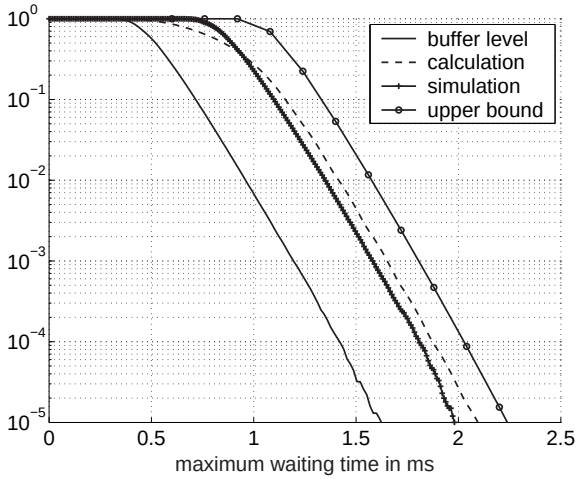


Figure 3.14: $\widehat{W}$ CCDF at the node level with CB-WFQ: $C = 20$ Mbps, $\Phi = 50\%$, $K = 60$, $M = 200$ byte, and $T = 20$ ms.

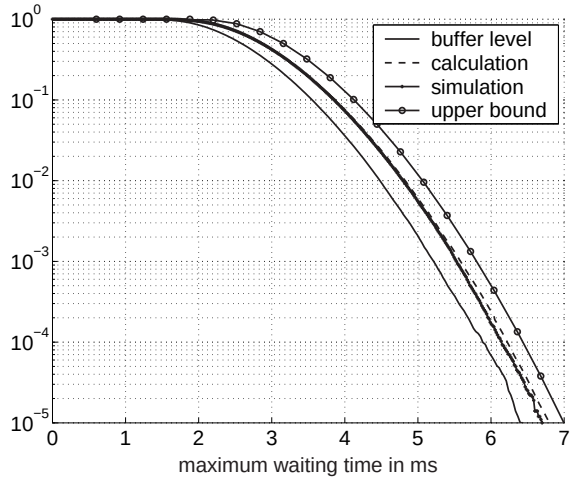
Obviously, w_{residue} can be considered to follow a uniform distribution in the interval $[0, T_{\text{MTU}}]$. As a result (3.19)–(3.21) still apply to the case of CB-WFQ. To validate their applicability, we perform simulations where the node model is implemented with CB-WFQ as the employed scheduling scheme. We compare results to those obtained using (3.19)–(3.21). In Figure 3.15, we plot the different curves obtained and they are the lower bound curve (refer to (3.19)), the calculation curve (refer to (3.21)), the upper bound curve (refer to(3.20)) and the simulation curve. Similar to the case of non-preemptive PQ, the simulation curve falls in between the two bounds and the calculation curve falls below the simulation curve at high probability values.



(a) Low voice load: $\Phi = 0.9$.



(b) Medium voice load: $\Phi = 0.5$.



(c) High voice load: $\Phi = 0.25$.

Figure 3.15: $\widehat{W}$ CCDF at the node level with CB-WFQ: $C = 20$ Mbps, $M_{\mathrm{MTU}} = 1500$ bytes, $M = 200$ bytes, and $T = 20$ ms.

3.4.4 Analysis at the Path Level

In this section, we extend the results from the node model to the path model so to include multiple hops between the source-destination pair. The maximum waiting time $\widehat{W}_{(s)}$ experienced by a voice packet among all packets along a given path s can be computed as

$$\widehat{W}_{(s)} = \sum_{l \in \mathcal{L}_s} \widehat{W}_l, \tag{3.23}$$

where $\widehat{W}_l$ is the maximum waiting time experienced at link l, $\forall l \in \mathcal{L}_s$ where $\mathcal{L}_s$ denotes the set of all links constituting path s. Based on the assumption of mutually independent hops [MWKB99] [Kle75] the path maximum waiting time distribution can be obtained by performing convolution of the maximum waiting time experienced at each link along path s, i.e.,

$$\mathrm{F}\left(\widehat{W}_{(s)}\right) = \Xi_{l \in \mathcal{L}_s} \mathrm{F}\left(\widehat{W}_l\right), \tag{3.24}$$

where Ξ denotes the convolution operation. Doing so, the distribution of $\widehat{W}_{(s)}$ is obtained and then we can solve for a capacity C_{voice} using the Almost Guaranteed QoS method. Having just one equation, we can solve for only one unknown that can be a common C_{voice} along the whole path. However, it can definitely happen that different number of active connections are routed through the individual links of the path and thus each would require a different value for C_{voice}. Knowing this, we are left with one equation having multiple unknowns (amounting to L_s where L_s denotes the number of links constituting path s). The computation of the capacities of the different links along the path can then be formulated as an optimization problem that searches for the combination of capacities that provides the optimal tradeoff between a defined cost and the required communication quality.

To advance from the node level to the path level, quality of service parameters should be transformed as well. With respect to the delay budget criterion, there exists a linear relation between the node delay budget and the path delay budget where the path delay budget is simply a linear addition of the individual node delay budgets. However, in regards to the outage probability, a more complex relation exists among the outage probabilities of the different detail levels of Figure 3.1. As a result, we propose a simplified mapping that transforms P_{out} from the node level to the path level. If the link outage probability $P_{\text{out,link}}$ is given, the path outage probability $P_{\text{out,path}}$ can be derived in two ways as demonstrated below.

1. To guarantee satisfactory performance, all hops along a flow path should comply with their respective QoS constraints. This is, however, a tight assumption and so a relatively low $P_{\text{out,path}}$ is expected. For a flow path consisting of L hops, we can formulate the following based on the assumption of mutually independent hops.

$$\begin{aligned} 1 - P_{\text{out,path}} &\triangleq \mathrm{P}\{(\widehat{W}_1 \leq \widehat{\mathcal{D}}_{\text{link}}) \cap (\widehat{W}_2 \leq \widehat{\mathcal{D}}_{\text{link}}) \cap \ldots \cap (\widehat{W}_L \leq \widehat{\mathcal{D}}_{\text{link}})\}, \\ &= (1 - P_{\text{out,link}})^L. \end{aligned} \tag{3.25}$$

Therefore, the first option (option 1) yields

$$P_{\text{out,path}} = 1 - (1 - P_{\text{out,link}})^L. \tag{3.26}$$

2. For a given hop, we assume that QoS constraints at other hops are met and so quality degradation is observed when the constraints are violated at this hop. This is rather a looser assumption than the former but yields simpler mapping. Then, for a flow path of L hops, we can similarly formulate the following based on the assumption of mutually independent hops

$$\begin{aligned} P_{\text{out,path}} &\triangleq \mathrm{P}\{(\widehat{W}_1 > \widehat{\mathcal{D}}_{\text{link}})|(\widehat{W}_2 \leq \widehat{\mathcal{D}}_{\text{link}}), \ldots, (\widehat{W}_L \leq \widehat{\mathcal{D}}_{\text{link}})\}, \\ &= P_{\text{out,link}}. \end{aligned} \tag{3.27}$$

Therefore, the second option (option 2) infers a direct mapping,

$$P_{\text{out,path}} = P_{\text{out,link}}. \tag{3.28}$$

Comparison Results and Analysis The two options for computing the outage probability P_{out} needed for the new capacity assignment procedure are investigated and compared against each other by means of the following example.

Example 3.2 *A stream of $K = 50$ active voice connections is routed on a 3-hop path ($L = 3$) in an IP network. Each connection generates $M = 200$-byte IP packet every $T = 20$ ms. Propagation delay is neglected and the maximum transfer unit of each link has a size of $M_{\text{MTU}} = 1500$ bytes. Calculate the link capacity needed along the path to ensure a high quality end-to-end communication. The network delay threshold is required to be $\widehat{\mathcal{D}}_{\text{net}} = 15$ ms and an outage probability of at most $P_{\text{out}} = 10^{-5}$ is allowed, i.e.,*

$$\mathrm{P}\{\widehat{W}_1 + \widehat{W}_2 + \widehat{W}_3 \geq 15\} \leq 10^{-5}.$$

Since all links of the 3-hop path carry equal traffic load, an identical capacity value for serving voice traffic is assumed on all links. The two options presented earlier for computing the outage probability at the path level are used to compute the needed capacity values. The obtained capacity values are then compared against each other and the most appropriate option to compute the outage probability is then selected for further analysis. Table 3.2 shows the results for different scheduling schemes: preemptive and non-preemptive PQ where the voice traffic class is assigned the highest priority, and CB-WFQ where the voice traffic class is given $\Phi = 50\%$ share of link capacity. *Per-hop ($P_{\text{out}} = 0$)* column of Table 3.2 shows capacity results calculated according to the worst-case capacity assignment method on a per-hop basis where a deterministic delay limit of 5 ms is strictly guaranteed on each link.

Per-hop ($P_{\text{out}} \neq 0$) refers to computing the link capacity based on the Almost Guaranteed QoS method applied on a per-hop basis. *End-to-End ($P_{\text{out}} \neq 0$)* refers to computing the link capacity based on the Almost Guaranteed QoS method applied on an end-to-end basis. For *Per-hop ($P_{\text{out}} \neq 0$)*, both mapping options of the outage probability are performed leading to varying per-hop outage probabilities. In Table 3.2, P_{out} is presented as a per-hop value in both options of *Per-hop ($P_{\text{out}} \neq 0$)* and in *Per-hop ($P_{\text{out}} = 0$)*, and as an end-to-end value in *End-to-End ($P_{\text{out}} \neq 0$)*. Comparing capacity results of *Per-hop ($P_{\text{out}} \neq 0$)* to the more accurate *End-to-End ($P_{\text{out}} \neq 0$)*, Table 3.2 shows that both mapping options lead to safe dimensioning, however, option 2 yields more economical results (lower capacity values). Hence, option 2 can be used to perform the mapping required for *Per-hop ($P_{\text{out}} \neq 0$)*. Thus, $P_{\text{out,path}}$ is set to P_{out} and will be referred to as P_{out} in the sequel. If results of option 2 of *Per-hop ($P_{\text{out}} \neq 0$)* are compared to those of *Per-hop ($P_{\text{out}} = 0$)*, it is clearly shown that the required capacity can be greatly reduced for all scheduling schemes when a slight degradation in quality is tolerated. ■

Table 3.2: Results of link capacity calculations (in Mbps) based on various capacity assignment approaches.

	Per-hop ($P_{\text{out}} \neq 0$)		*End-to-End*	*Per-hop*
	Option 1	Option 2	($P_{\text{out}} \neq 0$)	($P_{\text{out}} = 0$)
P_{out}	3.3×10^{-6}	10^{-5}	10^{-5}	0
Preemptive PQ	$C = 5.09$	$C = 4.96$	$C = 4.20$	$C = 16$
Non-preemptive PQ	$C = 6.17$	$C = 6.04$	$C = 4.96$	$C = 18.4$
CB-WFQ ($\Phi = 0.5$)	$C = 11.07$	$C = 10.83$	$C = 9.12$	$C = 34.4$

In the above example, we solved for one value of C_{voice} that is assumed to be fixed all through the individual links of a given path. However, in the case of different number of active connections routed through these links, it is no more logical to set one value for C_{voice} throughout the path. We need to find a set of values for C_{voice} on the different links of the path and this can be done by means of optimization techniques that search for the optimal set of values that minimizes the total cost. In the following section, we formulate this problem into an objective function and a number of constraints to solve it as an optimization problem.

3.4.4.1 Problem Formulation

In this section, we address the problem of optimal capacity assignment for voice along a network path where the total link capacity shares allocated to voice traffic are minimized subject to performance constraints in terms of delay requirements. Doing so, we introduce a new extension to traditional capacity assignment problems by setting a statistical bound on the maximum delay experienced among all voice packets.

Our CA problem can be stated as follows.

Problem $(\mathcal{P})$

$$Z = \min \left(\sum_{l \in \mathcal{L}_s} C_{\text{voice},l} \right), \tag{3.29}$$

subject to

$$\mathrm{P}\left\{ \widehat{W}_{(s)} > \widehat{\mathcal{D}}_{\text{path}} \right\} \leq P_{\text{out}}, \tag{3.30}$$

$$C_{\text{voice},l} \geq K_l \cdot r \quad \forall l \in \mathcal{L}_s, \tag{3.31}$$

where $\widehat{W}_{(s)}$ denotes the maximum waiting time along the links of a given path s, K_l is the number of active voice connections routed through link l, and $\mathcal{L}_s$ denotes the set of all links of path s.

The problem defined by equations (3.29)–(3.31) will be referred to as problem $(\mathcal{P})$. In problem $(\mathcal{P})$, (3.29) denotes the objective function which aims for minimizing the total network capacity cost over all links. The constraint in (3.30) represent the performance criterion which requires the maximum path delay among all packets belonging to the given traffic demand pair exceed the given threshold $\widehat{\mathcal{D}}_{\text{path}}$ only with an outage probability P_{out}. Furthermore, the required capacity $C_{\text{voice},l}$ of each link l has to be at least equal to the mean rate sum of the active number of

connections traversing l as the system might become instable otherwise. Recalling the results in Section 3.4.3, the maximum waiting time distribution for voice traffic is computed for a single link l to be a function g$(\cdot)$ of the number of active connections on link l, the link capacity $C_{\text{voice},l}$, the delay threshold $\widehat{\mathcal{D}}_l$, and finally the bit rate of each connection r, i.e.

$$\mathrm{P}\left\{\widehat{W}_l > \widehat{\mathcal{D}}_l\right\} = \mathrm{g}\left(K_l, C_{\text{voice},l}, r, \widehat{\mathcal{D}}_l\right). \tag{3.32}$$

As a result, we can note that constraints in (3.30) are non-linear and entail several numerical convolutions. This makes the problem more complicated.

3.4.4.2 Problem Complexity

We can solve problem $(\mathcal{P})$ using a Quasi-Newton method that is a gradient optimization method which is known to converge rather fast. However, it requires long computation time even for 3-link path due to the gradient calculations which are quite complex in this problem. This fact reveals all gradient methods as inappropriate for this type of problem. As an alternative solution, we can reformulate $(\mathcal{P})$ as an unconstrained minimization problem using the penalty function technique which is then solved by means of the Simplex method of Nelder and Mead (not to be confused with the Simplex method known for linear optimization) [Ber99]. Though the Simplex method is generally known to be inefficient as its theoretical convergence properties are often unsatisfactory. However, it is fairly simple to implement and does not require gradient calculations, the fact that makes it more convenient than other methods. Nevertheless, it also requires considerable computation time to solve $(\mathcal{P})$. In summary, the Quasi-Newton method proved inefficient due to gradient calculations in every iteration though it converges with a low number of iterations, while the Simplex method does not require gradient calculations but requires many iterations to converge where in every iteration, a convolution of the maximum waiting time distribution over all links of a communication path is performed. This brings up the need to reformulate problem $(\mathcal{P})$ so as to reduce the number of time-consuming operations.

3.4.4.3 Simplification through Link Decomposition and Solution Approach

In this section, we simplify $(\mathcal{P})$ in a way to avoid numerically computing multiple convolutions of different distributions. We define a new problem $(\mathcal{P}')$ based on link decomposition. Given an outage probability and a delay budget allocated to one link, the link capacity is computed independently from other links along the path. Thus, we define $(\mathcal{P}')$ as

Problem $(\mathcal{P}')$

$$Z = \min\left(\sum_{l\in\mathcal{L}_s} C_{\text{voice},l}\right), \tag{3.33}$$

subject to

$$\sum_{l\in\mathcal{L}_s} \widehat{\mathcal{D}}_l \;\leq\; \widehat{\mathcal{D}}_{\text{path}}, \tag{3.34}$$

$$C_{\text{voice},l} \;\geq\; K_l \cdot r \quad \forall l \in \mathcal{L}_s. \tag{3.35}$$

The constraint in (3.34) assure that the maximum path delay should not exceed a given delay threshold $\widehat{\mathcal{D}}_{\text{path}}$. Note that constraint (3.34) is linear and the new optimization variables are the

individual delays allocated to each of the links in the network. For each $\widehat{\mathcal{D}}_l$, $C_{\text{voice},l}$ is calculated by solving (3.32) numerically for $\mathrm{P}\left\{\widehat{W}_l > \widehat{\mathcal{D}}_l\right\} \leq P_{\text{out}}$, where $\widehat{W}_l$ is the maximum waiting time experienced at link l. If the same P_{out} were used in both problems, as an end-to-end value in $(\mathcal{P})$ and as a per-hop value in $(\mathcal{P}')$, $(\mathcal{P}')$ would yield a smaller end-to-end outage probability and thus represents a sub-optimal solution, yet a more conservative one as compared to $(\mathcal{P})$. At this point, $(\mathcal{P}')$ is reformulated into an unconstrained optimization problem using Multiplier and Lagrangian method that belongs to penalty function techniques and that constructs the new objective function in the form of Generalized Lagrangian function expressed as follows,

$$L_{\mathrm{G}}(\Gamma, \Lambda) = \sum_{l \in \mathcal{L}_s} C_{\text{voice},l} - \frac{1}{4\alpha}\left(\lambda_s^4 - \left(\lambda_s - \alpha\left(\widehat{\mathcal{D}}_{\text{path}} - \sum_{l \in \mathcal{L}_s} \widehat{\mathcal{D}}_l\right)\right)_+^4\right), \tag{3.36}$$

where Λ is the vector of Lagrange multipliers λ_s, α is a sufficiently large number and $(x)_+ = \max(x, 0), \forall x \in \mathbb{R}$. Note that constraints (3.35) are not included in the Generalized Lagrangian function L_{G} since it is obsolete at this point: in $(\mathcal{P}')$, the optimization variables are the link delay budgets which are fed into (3.32) to compute the link capacity and (3.32) assures implicitly that the link capacity is at least $K_l \cdot r$, $\forall l \in \mathcal{L}_s$. In $(\mathcal{P})$, on the other hand, the solution approach was different due to the fact that the link capacities were the optimization variables and so the constraint $C_{\text{voice},l} \geq K_l \cdot r$ has to be taken into consideration whenever a new set of link capacities is selected. Finally, $(\mathcal{P}')$ is solved by minimizing $L_{\mathrm{G}}(\Gamma, \Lambda)$ by means of the Simplex method of Nelder and Mead.

3.4.4.4 One-Path Example

In this section, we investigate the capacity assignment problem for one-link and two-link paths in order to gain insight about its fundamental properties.

We first consider an individual link with 10, 50 and 100 active connections. Each connection carries voice traffic generated by G.711 coders with 80 kbps bit rate (at the IP layer). Figure 3.16 plots the required capacity in terms of the link delay budget for each scenario. As expected, link capacity can be reduced if higher link delays are acceptable. Especially for low delay values (i.e. for very strict delay requirements) a slight increase in acceptable delay can lead to a significant decrease of required capacity. However, once the capacity has been brought down to the sum of the bit rates of the allowed number of active connections, no further gains are possible. This is due to equation (3.35), which guarantees that the system is stable. As a consequence, the capacity curve stays constant from a certain link delay budget on. It is interesting to note that links carrying a higher number of connections reach the minimum capacity at lower delay budgets.

To further investigate the above observations, we consider the two-link path depicted in Figure 3.17, which consists of two links carrying K_1 and K_2 active connections respectively. The delay budgets allocated to the links are denoted as $\widehat{\mathcal{D}}_1$ and $\widehat{\mathcal{D}}_2$, respectively. The path delay threshold is 10 ms and the outage probability is set to 10^{-3}.

Figure 3.18 plots the capacity sum of both links when $K_1 = 10$ and $K_2 = 50$ in terms of $\widehat{\mathcal{D}}_1$ ($\widehat{\mathcal{D}}_2 = 10$ ms $- \widehat{\mathcal{D}}_1$). The minimum value of the capacity sum is marked by a cross ($\times$) and it is achieved when links 1 and 2 are allocated $\widehat{\mathcal{D}}_1 = 4.3$ ms and $\widehat{\mathcal{D}}_2 = 5.7$ ms, respectively.

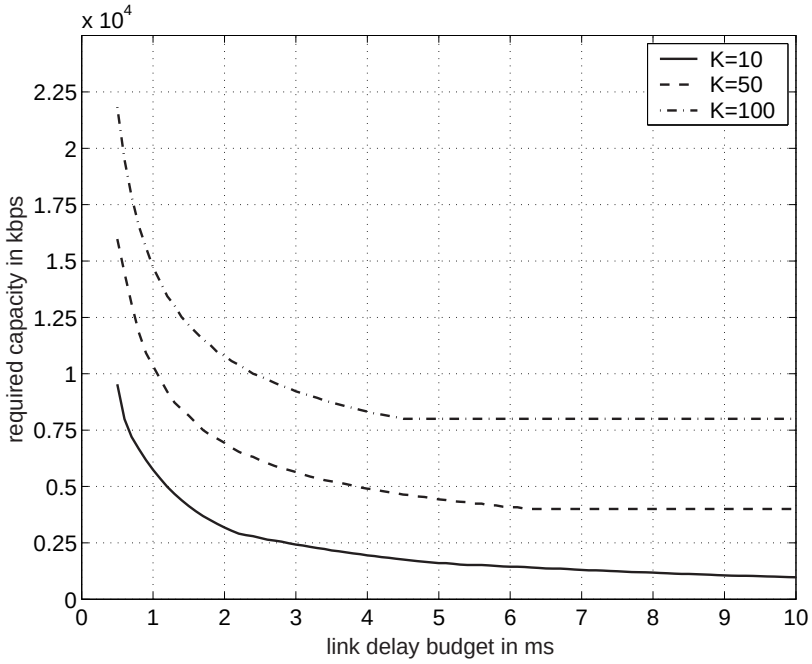


Figure 3.16: Required capacity in kbps versus link delay budget $\widehat{\mathcal{D}}_{\text{link}}$ in ms ($P_{\text{out}} = 10^{-3}$).

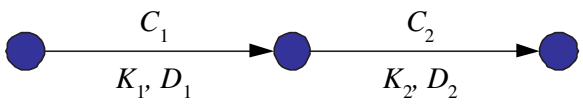


Figure 3.17: Two-link path model.

Figure 3.19 plots a set of the total capacity sum curves, corresponding to various K_1 and K_2 values, in terms of the delay budget allocated to link 1. The minimum capacity sum value is marked by a cross and multiple values are marked by a small box around them. Examining Figure 3.19, the following observations can be drawn.

- If $K_1 = K_2$, links 1 and 2 are assigned equal delay values. Such observation is logical as both links are carrying equal number of connections. It is important to note that this observation also holds for $K_1 = K_2 = 100$. However, in this case the minimum capacity of a link is already reached for a delay budget of 4.6 ms. Therefore, the capacity sum is minimal for any values $\widehat{\mathcal{D}}_1$ and $\widehat{\mathcal{D}}_2$ with $4.6 \text{ ms} < \widehat{\mathcal{D}}_1, \widehat{\mathcal{D}}_2 < 5.4$ ms.
- If $K_1 = 10$ and $K_2 = 50$, link 1 requires less delay than link 2. This observation is intuitive as one would directly expect that higher number of connections require more time to be served.
- If $K_1 = 10$ and $K_2 = 100$, link 1 requires more delay that link 2. This sounds logical when one realizes that link 2 already requires its minimum capacity when it gets a delay budget of at least 4.6 ms, and so the extra delay can be allocated to link 1. The same reasoning applies to the case when $K_1 = 50$ and $K_2 = 100$.

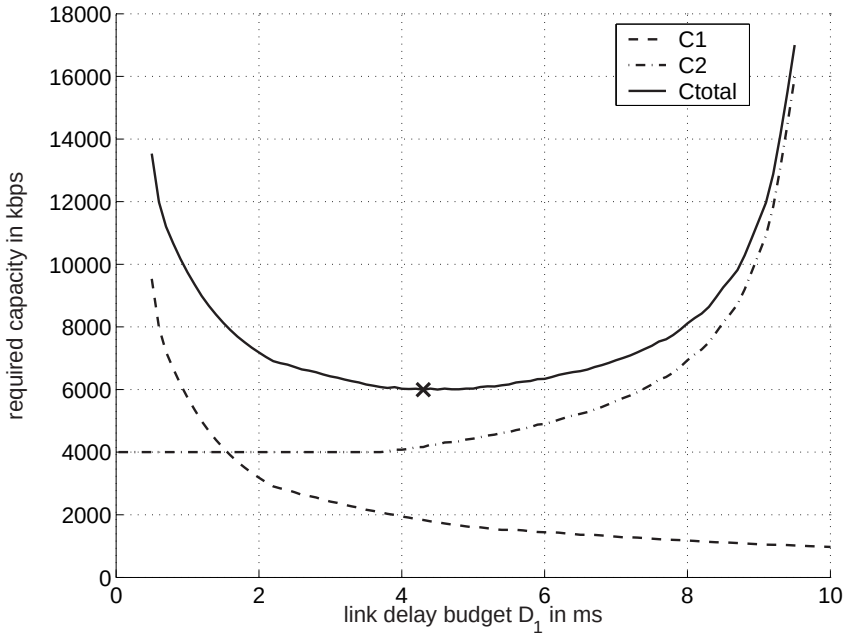


Figure 3.18: Individual link capacities and capacity sum versus $\widehat{\mathcal{D}}_1$.

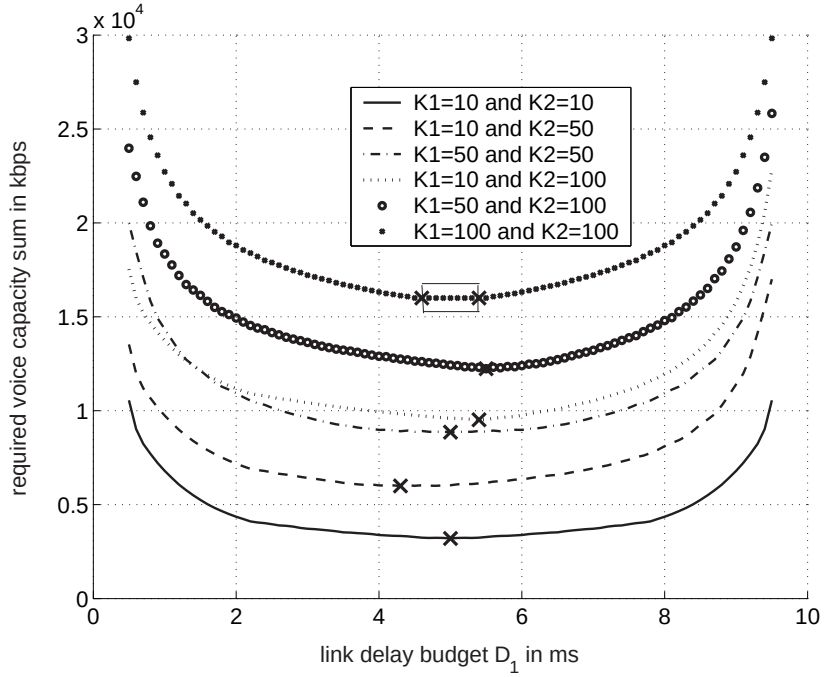


Figure 3.19: Capacity sum of links 1 and 2 versus $\widehat{\mathcal{D}}_1$.

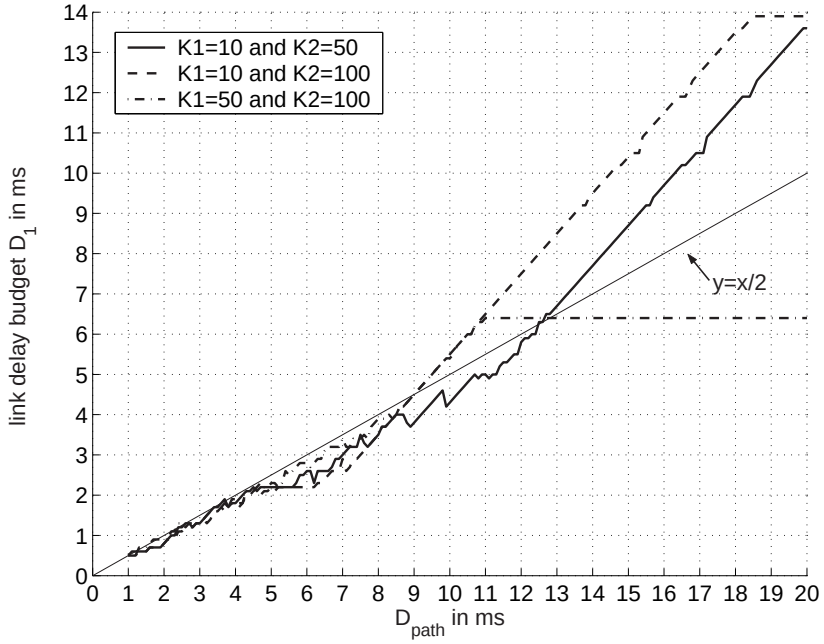


Figure 3.20: $\widehat{\mathcal{D}}_1$ versus $\widehat{\mathcal{D}}_{\text{path}}$.

It is interesting to monitor how the end-to-end delay budget is allocated between the two links so as to achieve the minimum capacity sum. Figure 3.20 plots the values of $\widehat{\mathcal{D}}_1$ that result in the minimum network costs as the path delay threshold varies. Examining the figure, we can make the following observations. We note that while these observations correspond to the curve of $K_1 = 50$ and $K_2 = 100$, they also apply to other combinations of K_1 and K_2 values in a similar way.

- At low $\widehat{\mathcal{D}}_{\text{path}}$ (up to 4.7 ms), the curve almost overlaps with $y = x/2$ line. Hence, the delay budget is allocated equally to both links irrespective of the number of connections on each of the links.

- If $\widehat{\mathcal{D}}_{\text{path}}$ falls between 4.7 ms and 9 ms, link 1 is allocated less delay than link 2, i.e., the link with more connections is granted more delay. This is more obvious in the case of $K_1 = 10$ and $K_2 = 100$.

- If $\widehat{\mathcal{D}}_{\text{path}}$ falls between 9 ms and 11 ms, link 1 is allocated more delay than link 2, i.e. the link with fewer connections is granted more delay. This is explained by the fact that link 2 is already capable of serving all its connections with the minimum capacity in 4.6 ms of delay. The slope of this part of the curve gets back to 1 again as all the delay above the 4.6 ms is totally allocated to link 1 to reduce its needed capacity and accordingly the total capacity.

- If $\widehat{\mathcal{D}}_{\text{path}}$ exceeds 11 ms, then both links reach their minimum capacity. Hence, any delay budget above the 11 ms has no extra advantage in decreasing the total costs.

3.4.5 Analysis at the Network Level

This section deals with generalizing the results of Section 3.4.4 to the network level where multiple paths exist. Our goal is also to find an optimal set of voice capacity shares of the complete list of links in the network.

3.4.5.1 Network Model

We consider an IP network with fixed physical network topology (node locations and connectivity among them). $\mathcal{L}$ denotes the set of unidirectional links. The projected traffic demand of interactive voice service is indicated in Erlang, in the form of origin-destination (OD) pairs, and is represented by a matrix $\mathcal{A}$ where $\mathcal{A}[i,j] = a_{ij}$ denotes the traffic demand directed from node i to node j where V denotes the total number of source and destination nodes:

$$\mathcal{A} = \begin{bmatrix} 0 & a_{12} & \dots & a_{1V} \\ a_{21} & 0 & \dots & a_{2V} \\ \dots & \dots & \ddots & \dots \\ a_{V1} & a_{V2} & \dots & 0 \end{bmatrix}. \tag{3.37}$$

In the sequel, we refer to the set of nonzero traffic demand elements in $\mathcal{A}$ as $\mathcal{S}$. Though offered traffic demand values are estimates, the actual traffic demand may dramatically deviate from the given values if no control mechanism is employed in the network. To avoid any traffic deviation inside the network, we assume a pipe model where virtual connections are established between origin-destination pairs where call admission control is applied for each origin-destination pair.

Based on a given call blocking probability, the maximum number of active channels $\widehat{K}$ between one OD pair can be computed by inversely solving the Erlang B equation:

$$\widehat{K} = arg\left\{B = \mathrm{E}\left(a, \widehat{K}\right)\right\}, \tag{3.38}$$

where B denotes the maximum allowed blocking probability, E the Erlang B function, and a the offered traffic demand between the given OD pair. At this stage, the carried traffic demand is represented in terms of the maximum number of channels that can coexist between each OD pair. Each channel represents one voice connection. Furthermore, we consider nonbifurcated routing only where all traffic from a single OD pair follows the same path. With this assumption, we are relating each traffic demand with a single path. The maximum number of connections routed on link $l \in \mathcal{L}$ is denoted as $\widehat{K}_l$ and is computed by applying the given routing scheme.

3.4.5.2 Problem Formulation

At the network level, the CA problem becomes:

Problem $(\mathcal{P})$

$$Z = \min\left(\sum_{l\in\mathcal{L}} C_{\mathrm{voice},l}\right), \tag{3.39}$$

subject to

$$\mathrm{P}\left\{\widehat{W}_{(s)} > \widehat{\mathcal{D}}_{\mathrm{net}}\right\} \leq P_{\mathrm{out}} \quad \forall s \in \mathcal{S}, \tag{3.40}$$

$$C_{\mathrm{voice},l} \geq \widehat{K}_l \cdot r \quad \forall l \in \mathcal{L}, \tag{3.41}$$

where $\widehat{W}_{(s)}$ denotes the maximum waiting time along the links of a given path s, $\mathcal{S}$ denotes the set of all paths in the network, $\widehat{K}_l$ denotes the maximum number of active voice connections routed through link l, and $\mathcal{L}$ denotes the set of all links in the network. We note that at the network level, we consider $\widehat{K}$ rather than K in constraints (3.41). This is because the network is usually controlled by means of call admission control that limits the number of active connections admitted into the network by a maximum value $\widehat{K}$.

Constraints in (3.40) represent the performance criterion which requires that the maximum network delay of all packets belonging to traffic demand pair $s, \forall s \in \mathcal{S}$, exceeds the given threshold $\widehat{\mathcal{D}}_{\text{net}}$ only with an outage probability P_{out}. The required capacity $C_{\text{voice},l}$ of each link l has to be at least equal to the bit rate sum of the maximum number of connections traversing l.

3.4.5.3 Problem Simplification through Link Decomposition and Solution Approach

Due to the same complexity reasons presented in Section 3.4.4, problem $(\mathcal{P})$ is simplified by means of link decomposition and reformulated into problem $(\mathcal{P}')$ defined as follows.

Problem $(\mathcal{P}')$

$$Z = \min \left(\sum_{l \in \mathcal{L}} C_{\text{voice},l} \right), \tag{3.42}$$

subject to

$$\sum_{l \in \mathcal{L}_s} \widehat{\mathcal{D}}_l \leq \widehat{\mathcal{D}}_{\text{net}} \quad \forall s \in \mathcal{S}, \tag{3.43}$$

$$C_{\text{voice},l} \geq \widehat{K}_l \cdot r \quad \forall l \in \mathcal{L}. \tag{3.44}$$

Constraints in (3.43) constitute a set of constraints, each corresponding to one traffic demand. As some traffic demands may follow completely independent paths from those followed by other traffic demands, $(\mathcal{P}')$ may be divided into different sub-problems whose solutions make up the final solution. This will yield significant reduction in running time especially when multiprocessing is supported. $(\mathcal{P}')$ is reformulated again into an unconstrained optimization problem using Multiplier and Lagrangian method and the new objective function is constructed in the form of Generalized Lagrangian function as shown in (3.45). This function is then solved using the Simplex method of Nelder and Mead as done previously at the path level.

$$L_{\text{G}}(\Gamma, \Lambda) = \sum_{l \in \mathcal{L}} C_{\text{voice},l} - \frac{1}{4\alpha} \sum_{s \in \mathcal{S}} \left(\lambda_s^4 - \left(\lambda_s - \alpha \left(\widehat{\mathcal{D}}_{\text{net}} - \sum_{l \in \mathcal{L}_s} \widehat{\mathcal{D}}_l \right) \right)_+^4 \right). \tag{3.45}$$

3.4.5.4 Mesh Network Examples

The solution approach presented earlier is applied on a sample network scenario depicted in Figure 3.21 and denoted as $N11$. The network scenario represents an enterprise backbone network consisting of 11 nodes and 48 unidirectional links. At each node, 1000 users are connected where each user is expected to generate voice traffic load of either 0.1 Erlang or 0.2 Erlang in the busy hour. Each node communicates with five other nodes selected randomly.

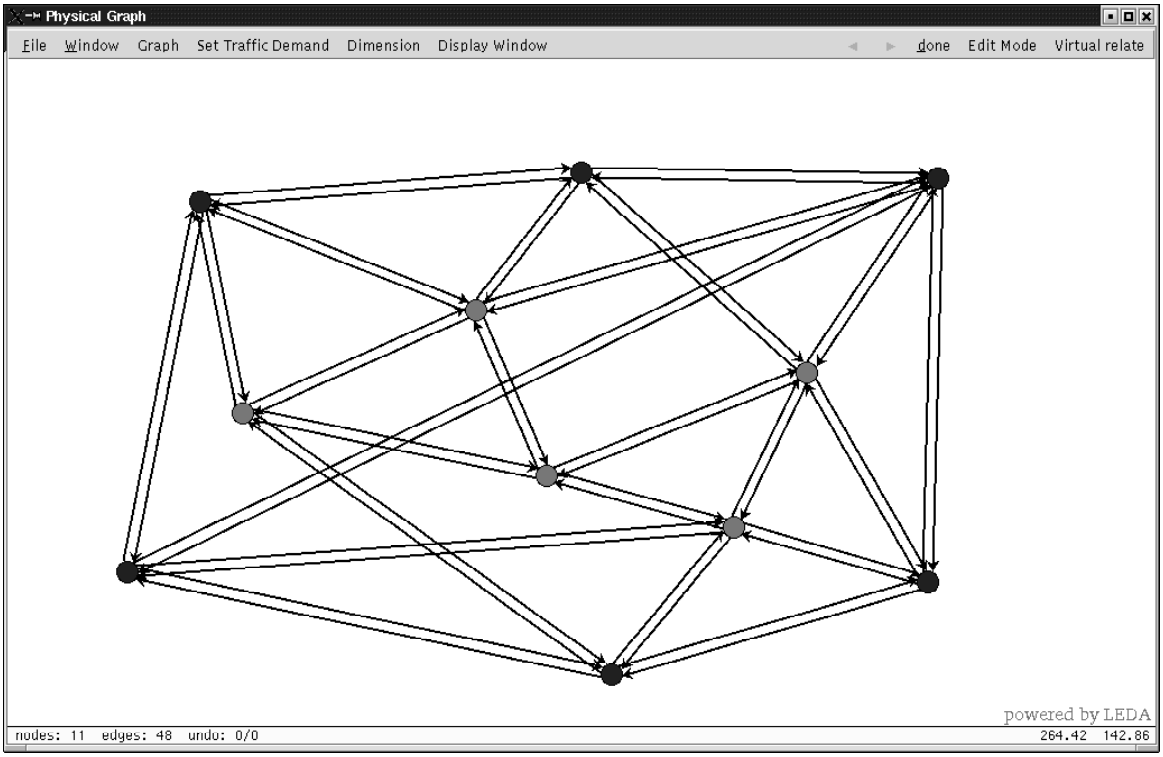


Figure 3.21: $N11$ network scenario.

Solving $(\mathcal{P}')$ as described in Section 3.4.5.3, the capacity shares required to serve the voice traffic on all links l of network $N11$ with minimum cost are evaluated. $(\mathcal{P}')$ can be compared to other approaches namely: *erlang* calculations, *Per-hop* ($P_{\text{out}} = 0$) calculations, and *Per-hop* ($P_{\text{out}} = 10^{-3}$) calculations. The first approach refers to the classical way of dimensioning used in circuit-switched networks and it is based on Erlang formulae which only considers the desired call blocking probability without any concerns about the traffic delay from source to destination. The *Per-hop* approaches, on the other hand, refer to calculations based on per-hop QoS constraints. For each link, the capacity is determined separately according to (3.32) where a mapping procedure transforms the network QoS value into a per-hop value. We assume that the network delay constraint $\widehat{\mathcal{D}}_{\text{net}}$ is partitioned into equal per-hop delay constraints, (i.e. $\widehat{\mathcal{D}}_{\text{net}}$ is divided by the hop-count of the longest path in the network) and that the per-hop outage probability is set equal to the network outage probability values as a conservative way of mapping (refer to Section 3.4.4). *Per-hop (*$P_{\text{out}} = 0$*)* is the worst-case capacity assignment approach, which results in hard guaranteed QoS where all packets are assured to be transmitted in a timely manner within a given strict delay constraint. Table 3.3 presents the capacity sum $\sum_{l \in \mathcal{L}} C_{\text{voice},l}$ obtained by each approach. The table shows that $(\mathcal{P}')$ is the closest to the erlang approach. It only requires an additional 3% of the total capacities to provide delay guarantees to the active voice connections in this example. Contrary to that, the *Per-hop (*$P_{\text{out}} = 0$*)* approach requires an additional capacity of 500% in order to provide hard delay guarantees. $(\mathcal{P}')$ is also shown to outperform the *Per-hop (*$P_{\text{out}} = 10^{-3}$*)* approach that requires 12.5% of additional capacity.

Table 3.3: Comparison of different capacity assignment approaches performed at the network level.

	erlang	problem $(\mathcal{P}')$	*per-hop*	
			$P_{\text{out}} = 0$	$P_{\text{out}} = 10^{-3}$
$\sum_{l \in \mathcal{L}} C_{\text{voice},l}$ in Mbps	314.8	324.7	1889	354.3
Additional capacity w.r.t. *erlang*	0%	3%	500%	12.5%

Figure 3.22 shows the distribution of the number of links whose resulting capacity falls within 5 Mbps ranges starting from 0 Mbps. In the figure, the *per-hop* ($P_{\text{out}} = 0$) requires much more capacity for most of its links than the other methods. $(\mathcal{P}')$ is shown to offer the most efficient network with high quality guarantees as its results are the closest to those of *erlang*.

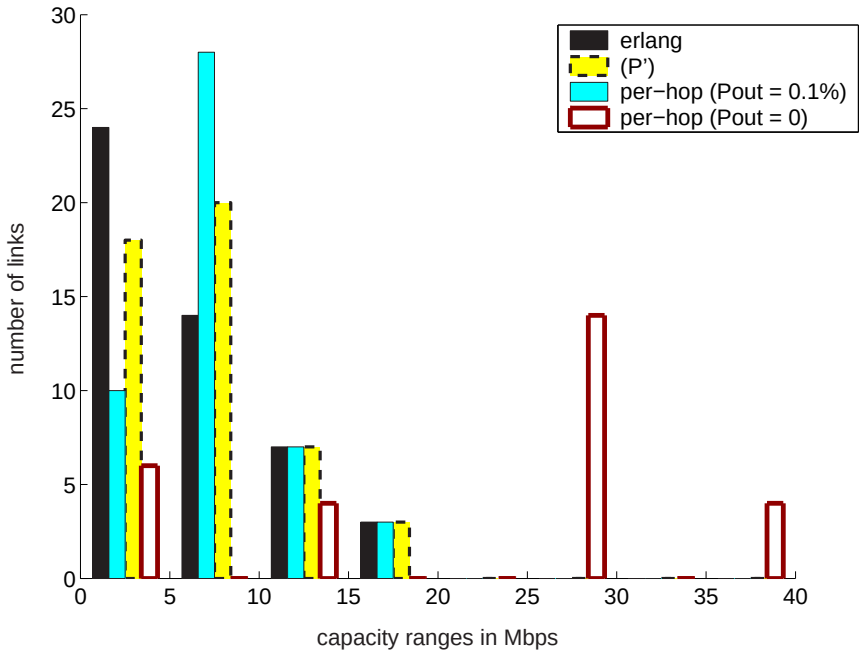


Figure 3.22: Distribution of resulting link capacities: $\widehat{\mathcal{D}}_{\text{net}} = 10$ ms.

As mentioned previously, *erlang* calculations can evaluate the maximum number of channels that can coexist on a given link. Traditionally, each connection requires only a service rate equal to the mean rate of the connection, thus, the capacity is set to $C_{\text{voice},l} = \widehat{K}_l \cdot r$. In our approach, however, $(\mathcal{P}')$ uses *erlang* calculations only to determine the maximum number of channels entering the network and then the link capacity is computed so as to achieve a statistical maximum delay bound among all packets of active connections. Based on this statistical maximum delay bound, the required capacity is no more $\widehat{K}_l \cdot r$ but larger. Figure 3.23 plots the normalized link capacity relative to r that is computed according to $(\mathcal{P}')$ for each link of the network. This actually represents the maximum number of channels that can be supported on each link in case the network is circuit-switched. Problem $(\mathcal{P}')$ shows slight overprovisioning (equivalent to 124 additional channels) for 21 links.

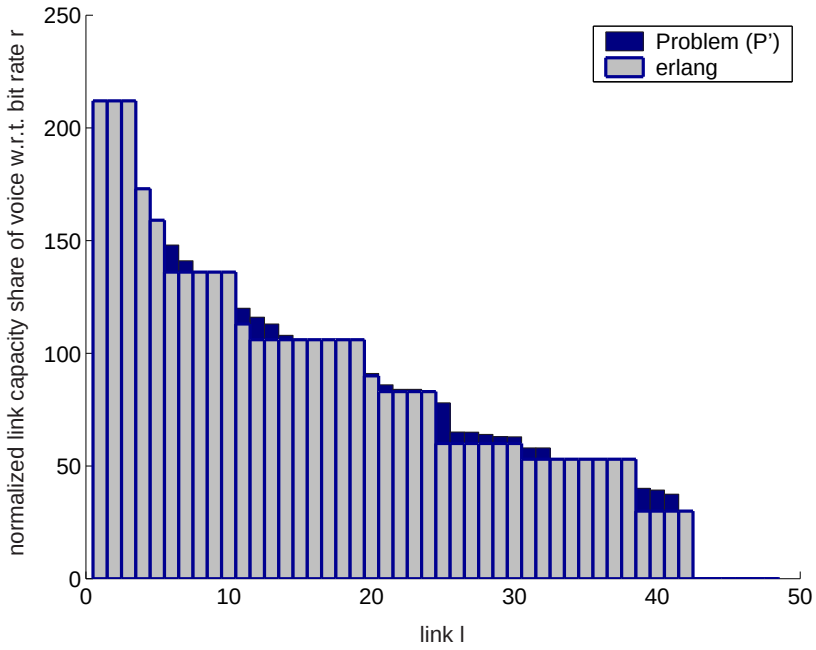


Figure 3.23: Maximum channel histogram of problem $(\mathcal{P}')$ as compared to *erlang* calculations.

3.5 Prototypical Implementation of a Generic Planning Tool

Traffic characteristics and requirements of network classes are quite different (refer to Chapter 2). This necessitates the need for various dimensioning schemes, each estimating the share of capacity needed for its corresponding class. In this section, we realize the given capacity planning problem in a dimensioning tool that supports different traffic classes belonging to the two broad traffic categories, namely stream (realtime) and elastic (non-realtime) traffic.

Traffic is roughly classified into two broad types: stream traffic and elastic traffic. Stream traffic is mainly generated by applications that establish realtime connections such as telephony and videoconferencing that are extremely sensitive to transmission delay. To meet the QoS criteria of stream traffic, stream connections require guaranteed transmission rate all through their activity period. Elastic traffic, however, allows for fluctuations in the transmission rate as long as the overall data throughput achieved at the end of the complete transmission is "acceptable". Elastic traffic is generated by traditional web applications, email, ftp, etc. Relevant QoS measures for elastic traffic type are packet loss and data throughput. As long as elastic traffic is carried by TCP connections, which assure reliable transmission including retransmission of lost packets, the main concern is focused onto the latter QoS measure, which is data throughput. Traffic of either elastic or stream nature can be further classified into finer granular categories (traffic classes) which maintain the same QoS measures but with different parameter values. Traffic classes are then mapped appropriately to the available network services provided by the actual technology of the network (e.g. integrated services (IntServ) model or differentiated services (DiffServ) model) to grant the associated traffic its desired QoS. For the development of the tool, we focus on the DiffServ architecture as it is more scalable and realistic and show in Figure 3.24 a sample mapping process [RBF02]. Each traffic class with a given traffic load

consumes part of the available link capacity. As a result, one can view the DiffServ link as a bundle of sub-links, each assigned a share of the total link capacity that is needed to achieve the requested QoS. According to the respective QoS measures, appropriate capacity assignment strategies are determined to adequately allocate network resources (determine sub-link capacities) for each class of traffic. Hence, the network planner should consider different capacity assignment methods for the different traffic classes.

3.5.1 Link Dimensioning Model

3.5.1.1 Dimensioning for Stream Traffic

Stream traffic is produced by a number of connections that require guaranteed transmission rate to assure a packet delay limit. Stream traffic is normally characterized by the offered traffic volume expressed in Erlang in addition to its traffic profile which can be represented in various ways such as the Token Bucket model [SRGT03]. The bit rate characteristics constituting the traffic profile can be transformed into a single value called the effective bit rate which is expressed in [Lin94] and [Lin99] as a function of mean rate, peak rate, link capacity, and packet loss. Based on the effective bit rate and traffic demand volume, the required transmission rate can be derived for a given blocking probability using known methods like [LH92] [NPGI99]. Dimensioning methodologies are traditionally proposed for ATM and circuit-switched networks where call blocking probability offered at the ingress of a network is the main QoS factor. In IP networks, however, packet delay represents a significant QoS factor which should be accounted for while planning link capacities. The worst-case CA approach provides stream traffic hard QoS guarantees with strict delay threshold on the expense of wasted network resources. This approach is used for comparison purposes. For this tool, however, we use the Almost Guaranteed QoS method proposed in Section 3.4. We note that the Almost Guaranteed QoS method assumes constant bit rate traffic and thus it can be applied to all real-time services generating constant bit rate (CBR) traffic. As to the case of voice transmission with the VAD (voice activity detection) feature enabled, the Almost Guaranteed QoS method can also be applied as argued previously in Section 3.2.4. While for the case of variable bit rate (VBR) video, one can

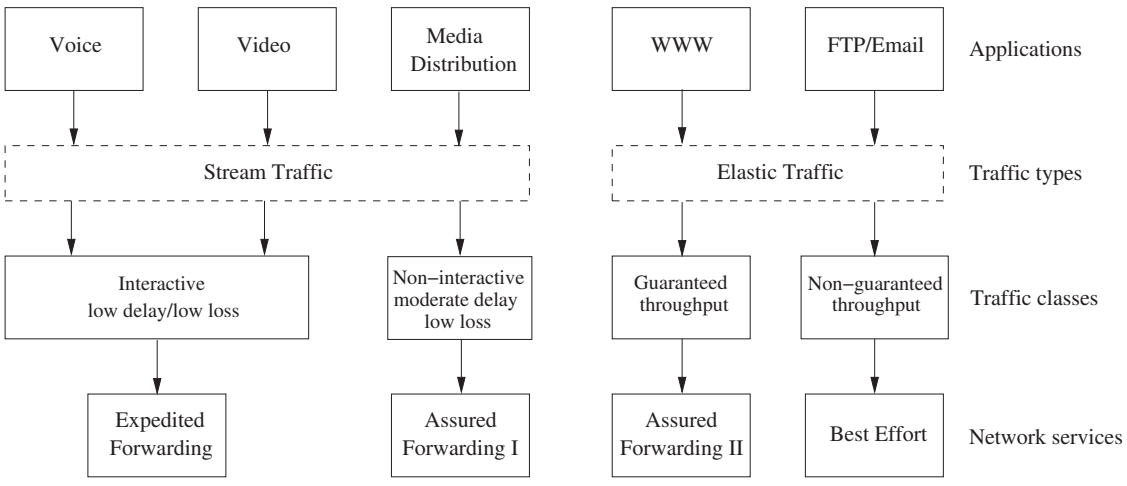


Figure 3.24: Sample mapping of applications to network services in DiffServ IP networks.

use a different method that is customized to this traffic type and its performance constraints. In Chapter 4, we introduce and investigate such a method.

3.5.1.2 Dimensioning for Elastic Traffic

An elastic traffic call corresponds to a file that needs to be transferred over the available network. In queuing theory perspective, this scenario represents a customer in a queuing system where customers arrive according to a stochastic process. Knowing that elastic traffic is normally handled by TCP connections, then with an ideal feedback mechanism, the service rate of the queuing system is shared equally among all customers in the queue according to a processor sharing discipline. The resulting system can then be modelled as an $M/G/R - PS$ queuing system whose sojourn time estimation can be used to evaluate the capacity share required to serve elastic traffic. Performance results are available in [Coh79]. In [Rie03], this dimensioning model is investigated and its applicability is verified by simulations where simulation results match the theoretical expectations closely. It is however noticed in [Rie03] that simulation and theoretical results mismatch when round trip times are rather long leading to underdimensioning. An extension of the method is then proposed to account for this factor leading to significant enhancement of the predicted theoretical results.

3.5.1.3 Dimensioning for Multiple Traffic Classes

In an integrated environment where traffic of stream and elastic characteristics share common resources, traffic engineering mechanisms (e.g. call admission control, traffic differentiation, and bandwidth reservation) are deployed to save the integrity of each traffic type. In order to assure a faultless service, an explicit evaluation of the minimum capacity requirement of each traffic class is therefore required. In our tool, we adopt the segregation approach of [NQBM99] to compute the total link capacity and this is done by a simple summation of the individual capacity shares of the available traffic classes. Mere summation of the individual capacity requirements might though lead to underutilized links since no multiplexing gain is considered. On the other hand, this approach provides a sort of protection to each traffic class against other classes in cases of congestion or ill-behaving sources, which generate high traffic load. Recent research investigates possibilities of dynamic resource allocation among different classes for an improved network utilization [FR01].

3.5.2 Generic Tool Architecture

For the ease of planning large-scale QoS-enabled IP networks, an architecture for a network dimensioning tool is proposed and examined in this section. The dimensioning strategies for stream and elastic traffic described earlier are applied to compute the required link capacities.

Given the network topology with the available link capacities, traffic characteristics, and demand matrices of each traffic class in addition to their QoS criteria, the task is to check whether the given network is well-qualified for the offered services and to compute the adequate capacity values for the network links to be capable of serving the carried traffic with the desired QoS constraints.

The proposed architecture of the network dimensioning tool is viewed as three layers, each handling different aspects of the dimensioning mechanism. The physical layer captures all properties of the given network related to topology configuration: connectivity of links and

nodes (switches/routers), scheduling scheme employed in the router nodes, link capacities, MTU sizes, etc. The traffic demand layer, however, represents the offered traffic demand of each class as a directed graph. We assume that the traffic demand of each supported traffic class is known and that call admission control is performed on a source-destination basis applying the pipe model so as to allow for no traffic deviation inside the network. If only coarse and general information about the ongoing traffic is available, we take up a rough way to estimate source-destination traffic demands: part of the traffic volume generated by one node is often directed to a group of interest and the rest of the traffic is homogenously been distributed to other nodes. The final layer in the architecture is an intermediary one and is called the virtual layer. It lies between the physical and traffic demand layers as an interconnection between both. It reads inputs from both layers, performs the necessary tasks for each traffic class, and finally hands the final combined results to the physical layer. The virtual layer carries virtual graphs, each associated with a traffic class. It maintains the same topology of the physical network at the IP layer while switched networks (parts of the network consisting of layer 2 switches only and connections among them) are replaced with virtual nodes. The virtual layer performs routing on each of its graphs and runs the corresponding dimensioning scheme to evaluate the needed capacity share of the considered class. This architecture is illustrated in Figure 3.25, which depicts the relations among the three layers.

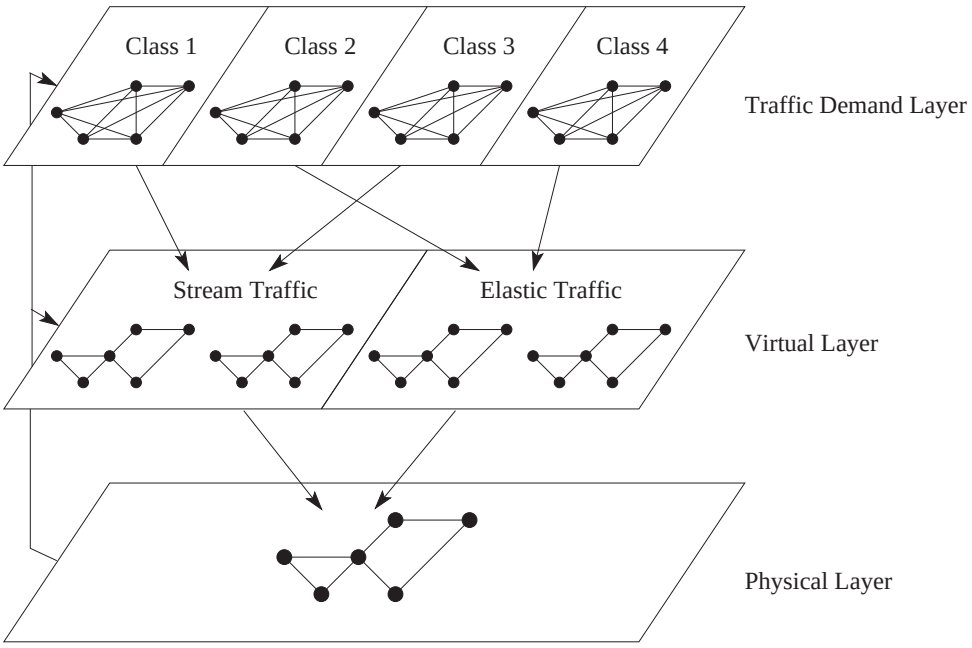


Figure 3.25: A layered structure for a network dimensioning tool.

3.5.3 Dimensioning Process

The flow of the dimensioning process is presented in the following steps.

1. Construct the physical graph from the given network configuration.

2. For each traffic class having a stream nature, the following capacity assignment procedure is performed.

   (a) Construct the traffic demand graph using the corresponding traffic demand matrix (given in Erlang).

   (b) Calculate the maximum number of simultaneously active connections $\widehat{K}$ using Erlang-B formula by allowing call blocking probability P_{block} on a source-destination basis applying the pipe model.

   (c) Construct the virtual graph as having the same topology of the physical graph at the IP layer while switched networks are represented by virtual nodes.

   (d) Using the employed routing scheme, identify the complete path of each traffic flow, given in the traffic demand graph, through the virtual graph. By summing up the number of active connections traversing each virtual link, we obtain the total number of connections that should be served simultaneously by each link within a delay constraint. The number of hops along the longest path is also recorded. We note that virtual nodes are ignored while counting the number of hops since layer 2 switches are assumed to contribute a negligible delay to the total network delay.

   (e) Compute the required transmission rate at each virtual link when allowing a given network outage probability $P_{\text{out,net}}$ and a delay threshold $\widehat{\mathcal{D}}_{\text{net}}$ from source to destination (both parameters are then transformed into per-hop values). While computing the capacity values, equal-rate coders are assumed.

      i. If $P_{\text{out,net}}$ is zero, use the worst-case capacity assignment approach to compute the required capacity values. For both PQ and CB-WFQ, the transmission rate R for this traffic class is calculated independently from other classes. For the case of CB-WFQ, however, a defined Φ is defined to limit the capacity share C_{class} assigned to this traffic class. The class capacity share C_{class} and the total link capacity C are thus computed as

$$C_{\text{class}} = \frac{\widehat{K} \cdot M + \Phi \cdot M_{\text{MTU}}}{\widehat{\mathcal{D}}_{\text{link}}}, \tag{3.46}$$

$$C = \frac{C_{\text{class}}}{\Phi}. \tag{3.47}$$

      ii. If $P_{\text{out,net}}$ is non-zero, the required capacity values are computed based on the new capacity assignment approach, the Almost Guaranteed QoS method. This method is applied on the network level and the optimization problem presented in Section 3.4.5 is solved as described. However, for a large network, it is recommended that a less complicated and time-consuming approach is used where dimensioning is performed on a per-hop basis. QoS parameters are mapped into per-hop values and then each link is treated separately and the method is applied link by link as if at the link layer. This way of treating the individual links separately is demonstrated in the example presented in Section 3.4.5.4 and is denoted as *Per-hop ($P_{\text{out}} \neq 0$)*. We note that the Almost Guaranteed QoS method is not suitable for variable bit rate video traffic due to reasons presented later in this dissertation and so a new method should be used for this purpose. The next chapter proposes a new capacity assignment method that is suitable for VBR video traffic.

iii. Links connecting a switch to a router are treated somewhat differently. They correspond to links connecting virtual nodes to a router at the virtual layer. Capacity computation of these links is illustrated by the example presented in Figure 3.26. $C_{\text{class},l}$ denotes the transmission rate allocated for the given traffic class on link l. $C_{\text{class},3}$ needed on link 3, which connects the virtual node (VN) to Router 3, is computed as

$$C_{\text{class},3} = \frac{\widehat{K}_1}{\widehat{K}_1 + \widehat{K}_3} \cdot C_{\text{class},1} + \frac{\widehat{K}_2}{\widehat{K}_2 + \widehat{K}_4} \cdot C_{\text{class},2},$$

where $C_{\text{class},1}$ and $C_{\text{class},2}$ are computed as described previously, and $\widehat{K}_i$, for $i = 1, \ldots, 4$, is the maximum number of flows traversing the associated links.

3. For each traffic class having an elastic nature, perform the following capacity allocation procedure.

   (a) Repeat steps 2a), 2c) and 2d). Note that the offered traffic demand in this case is given in bit rate, e.g. kbps, and that the maximum access rate r_{peak} of end users has to be set.

   (b) Use the $M/G/R - PS$ approach as described in [Rie03] for evaluating the needed capacity share required for the traffic class at hand.

4. Combine results of all traffic classes to obtain the total link capacity required.

   (a) If PQ is used at a link, the total capacity required at this link is

$$C = \sum_{q \in Q} C_{\text{class}_q}, \tag{3.48}$$

   where Q is the total number of available traffic classes and C_{class_q} is the computed capacity needed for traffic class q.

   (b) If CB-WFQ is used at a link, the required C at this link is

$$C = \max\left(\sum_{q \in Q} C_{\text{class}_q}, \max_{q \in Q} C \right), \tag{3.49}$$

   where C_{class_q} denotes the capacity share computed for traffic class q and C is the total link capacity computed in one virtual graph of traffic class q, $\forall q \in Q$.

   (c) The capacity of a switch-to-router link is obtained similarly as in step 4a).

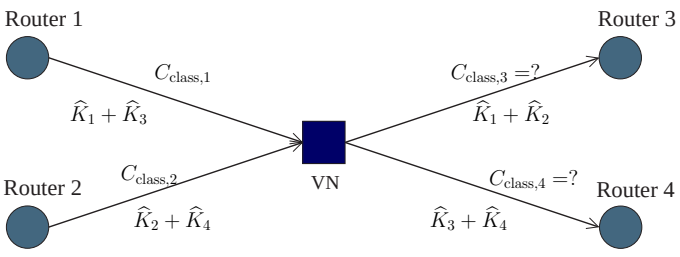


Figure 3.26: Dimensioning switch-to-router links.

3.5.4 Results and Analysis

A dimensioning tool is implemented using network planning library (NPL) [NPL02] and library for efficient data types and algorithms (LEDA) [LED02]. LEDA is a C++ class library available from Algorithmic Solutions Software and it provides relevant building blocks for managing objects such as graphs, sequences, dictionaries, etc. Based on LEDA, the Institute for Communication Networks at Munich University of Technology has developed NPL which consists of classes for handling nodes, edges and graphs and for enabling interconnections among elements of different LEDA graphs (layers). The NPL library is then developed as LEDA-independent with an enhanced user interface. The new version of NPL is called GRAPH library.

The dimensioning flow process is now applied on two sample scenario networks, $N5$ and $N50$ shown in Figure 3.27. Scenario $N5$ represents a WAN connecting five backbone routers; it consists of 5 nodes and 12 links. Scenario $N50$ is a larger-scale hierarchical network consisting of 50 nodes and 126 links. Each backbone router is connected to three or four access switches.

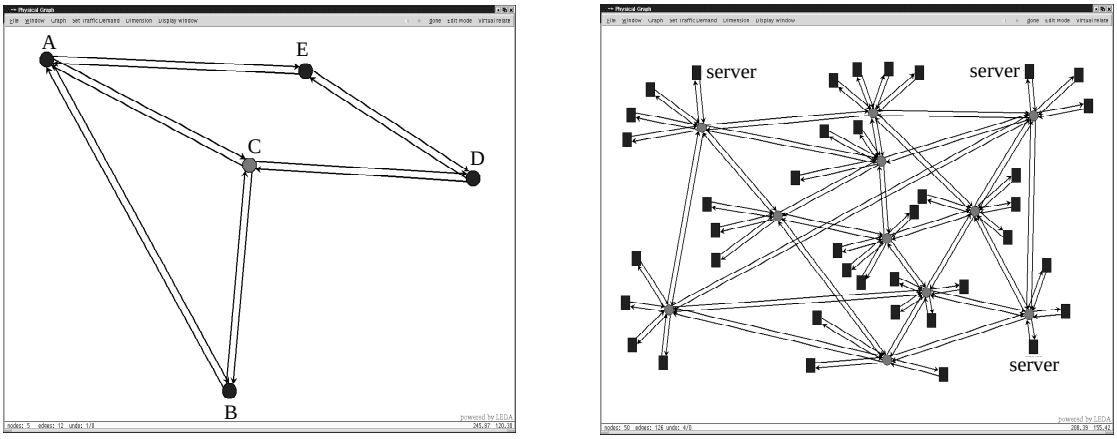


Figure 3.27: (a) Scenario $N5$: 5 nodes, 12 links. (b) Scenario $N50$: 50 nodes including 3 servers, 126 links.

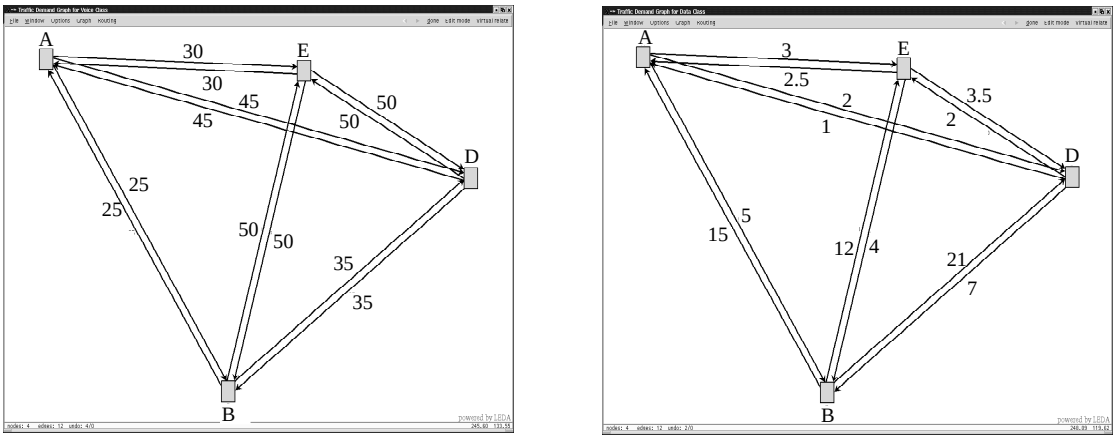


Figure 3.28: Scenario $N5$. (a) Voice traffic demand graph showing the traffic load in Erlang. (b) Data traffic demand graph showing the traffic load in Mbps.

Dark nodes in Figure 3.27 denote end nodes at which traffic statistics are available and they form the end elements of the traffic matrix. Figure 3.28 shows the traffic demand matrices of the two supported classes in scenario $N5$, voice and data, by means of traffic graphs. It is assumed that 1000 users in total are connected to each backbone node and each user generates 0.1 Erlang of voice traffic and 10 kbps of data traffic on average. Necessary input parameters including QoS criteria for the dimensioning process are the following.

Voice traffic class

- G.711 coder type is the used voice coder (packet size = 200 bytes, period = 20 ms)
- Non-preemptive PQ is used, voice class is set to highest priority, and $M_{\mathrm{MTU}} = 1500$ bytes
- $P_{\mathrm{block}} = 10^{-2}$
- $P_{\mathrm{out}} = 10^{-4}$
- Network delay threshold = 10 ms

Data traffic class

- Peak rate = 64 kbps
- Throughput = 80% of traffic mean rate

After reading in traffic information, carried traffic is calculated and distributed within the network according to OSPF (open shortest path first) routing at the virtual layer. Based on the resulting traffic distribution, capacity evaluation of each class is performed for all links of the virtual graphs. Voice and data virtual graphs are presented in Figure 3.29. Link capacities calculated by virtual graphs of voice and data are combined and handed in to the physical layer,

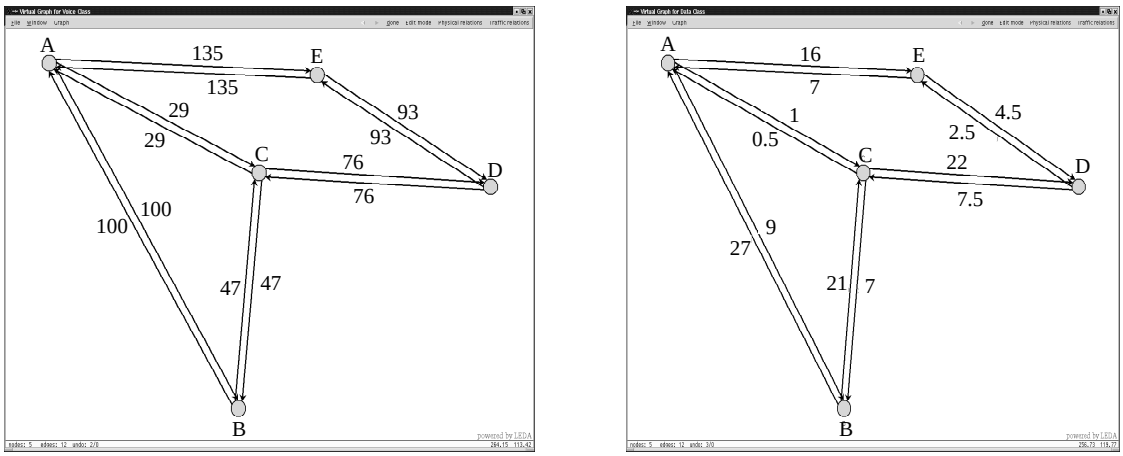


Figure 3.29: Scenario $N5$. (a) Voice virtual graph showing number of active connections routed on the links. (b) Data virtual graph showing the mean data rate in Mbps routed on the links.

which upgrades the physical network by specifying the required capacity values for all links. Results are summarized in Table 3.4. C_{data} and C_{voice} denote the capacity shares required for data and voice classes respectively. C_{voice} is computed using the Almost Guaranteed QoS method on a per-hop basis that is denoted as *Per-hop ($P_{\text{out}} = 10^{-4}$)*, the worst-case capacity assignment method that is denoted as *Per-hop ($P_{\text{out}} = 0$)*, and *Ratio* calculates the ratio of the values obtained by the latter method to those obtained by the former. It is obvious that *Per-hop ($P_{\text{out}} = 0$)* results in high capacity requirements as opposed to *Per-hop ($P_{\text{out}} = 10^{-4}$)* which provides a tradeoff solution by slightly softening QoS guarantees for the benefit of extensive reduction in capacity requirements. For example, *Per-hop ($P_{\text{out}} = 10^{-4}$)* results in a capacity of 11.0 Mbps for link AE which would be dimensioned by almost four times more (45.6 Mbps) if the worst-case CA method is used.

Table 3.4: Dimensioning results of network $N5$ (in Mbps)

			Per-hop ($P_{\text{out}} = 10^{-4}$)	*Per-hop ($P_{\text{out}} = 0$)*	Ratio
Source	Target	C_{data}	C_{voice}	C_{voice}	
A	E	16.2	11.0	45.6	4.1
E	A	7.2	11.0	45.6	4.1
A	B	9.2	8.8	34.4	3.9
B	A	27.2	8.8	34.4	3.9
A	C	1.1	4.5	11.7	2.6
C	A	0.6	4.5	11.7	2.6
B	C	21.2	5.6	17.4	3.1
C	B	7.2	5.6	17.4	3.1
C	D	22.2	7.3	26.7	3.7
D	C	7.7	7.3	26.7	3.7
D	E	2.7	8.3	32.2	3.9
E	D	4.7	8.3	32.2	3.9
Average		10.6	7.6	28.0	3.7

Following the same dimensioning procedure, network $N50$ is dimensioned assuming that 200 users are connected to each access node. Parameters given earlier for $N5$ are maintained except for the following.

Data traffic class

- Each user generates 22 kbps on average
- 54% of the generated traffic is directed in equal shares to given servers indicated in Figure 3.27b
- Remaining traffic is distributed evenly to all other end nodes (access switches)
- Upstream/downstream ratio to/from server is 2:5

Figure 3.30 shows the distribution of the resulting link capacities by counting the number of links whose dimensioned capacity falls within each of the shown 30 Mbps ranges starting from

0 Mbps. For demonstration, these collected statistics are taken for the backbone links only, i.e. links connecting access switches to backbone routers are excluded. Results are obtained using the two dimensioning methods, *per-hop* ($P_{\text{out}} = 10^{-4}$) and *per-hop* ($P_{\text{out}} = 0$). It is shown that *per-hop* ($P_{\text{out}} = 0$) requires much more capacity values for most of its links as compared to *per-hop* ($P_{\text{out}} = 10^{-4}$). For example, twelve links have their capacity values between 150 and 180 Mbps when *per-hop* ($P_{\text{out}} = 0$) is used while only two links have their capacity values in the same range when *per-hop* ($P_{\text{out}} = 10^{-4}$) is used.

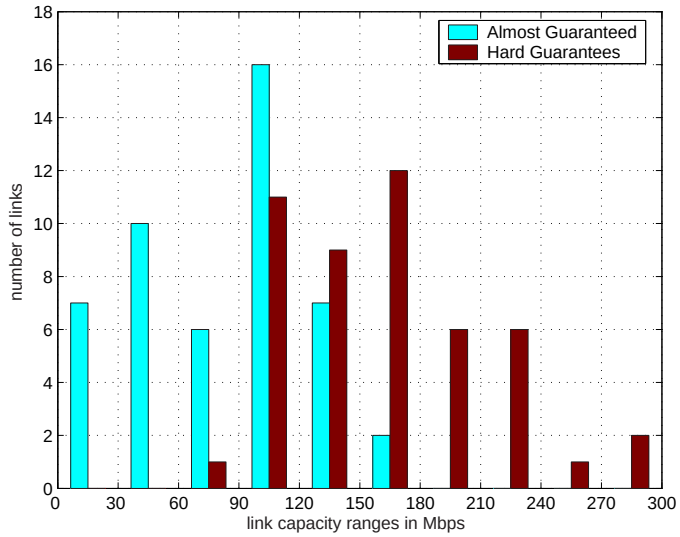


Figure 3.30: Dimensioning results of network $N50$ (in Mbps).

3.6 Summary

In this chapter, we addressed the problem of capacity assignment for the interactive voice service running over IP networks. Traditional capacity assignment techniques used for (virtual) circuit-switched networks do not account for the delay factor as it is nearly negligible in such networks. However, if delay constraints have to be strictly abided by for hard QoS guarantees, then huge capacity requirements are solicited making it unaffordable. To this end, we propose a novel tradeoff method and motivate it by means of simulations, which demonstrate reasonable capacity requirements for high performance level. Thereafter, we define the new method clearly in a step manner and proceed with mathematical analysis that allows us to compute the capacity requirements directly rather than by simulations. To start with, the mathematical analysis is performed based on a simplified model which we call the buffer model. The model is then developed gradually up until a general network model is reached. At this point, the capacity assignment problem is formulated into an optimization problem that aims for minimizing the total capacity values subject to nonlinear performance constraints. The method is finally realized in a generic network planning tool that is suitable for dimensioning large-scale multiservice IP networks. The tool is designed into a three-layered architecture where each layer is assigned a number of tasks independently from other layers. The tool is finally tested on two realistic network scenarios.

4

Network Dimensioning for Video Services with Statistical QoS Guarantees

Video communication services are a major capacity consuming applications for IP networks. VBR-coded video is preferred over CBR-coded video due to its advantages in statistical multiplexing gain and consistent video quality. Though, VBR traffic causes a serious challenge for network planners to assess the best tradeoff capacity share required to achieve high performance and low costs. In this chapter, we introduce a dimensioning model based on the waiting time distribution of video frames and a desired outage probability that defines the extent to which the given delay threshold is adhered to. Unlike voice communications, the maximum waiting time model is not suitable for link dimensioning for interactive VBR video traffic yet it is applicable for the CBR form of video traffic.

We begin with an overview on the common MPEG-coded traffic pattern in Section 4.1. In Section 4.2, we provide a brief description about few theoretical video models and assess their performance with respect to real video models so they can be used to drive network simulations. Later on, the realtime conditions of MPEG-coded video traffic over IP networks are considered in Section 4.3 and the dimensioning model is introduced in Section 4.4 where several factors affecting this model are handled and their influence is evaluated. Finally, Section 4.5 wraps up the chapter.

4.1 Overview on MPEG Video Encoding

MPEG (Moving Picture Expert Group) represents a family of standards used for coding audio-visual information in a digital compressed format. It has been the “de-facto” standard for video streaming and realtime video applications on the internet. This is due to its high compression ratios with minimal impact on the perceived quality.

The MPEG standard defines three different types of frames, each of which has its own properties and coding mechanisms. The frame types are named as such: “intra” picture (I-frame), “predictive” picture (P-frame) and “bidirectionally-predictive” picture (B-frame). The I-frame is coded independently of previous or future frames. I-frame coding is based on the frame itself only and it is similar to static JPEG image (i.e. intra-frame coding). The P-frame is coded based on forward prediction that is performed with respect to the past I-frame or P-frame. P-frame coding uses motion compensation prediction to provide more compression. Finally, the B-frame is coded based on motion compensated prediction as well from a forward and/or backward I-frame or P-frame. This technique is called bidirectional prediction and can thus provide the most compression. Figure 4.1 summarizes the different frames and their dependencies.

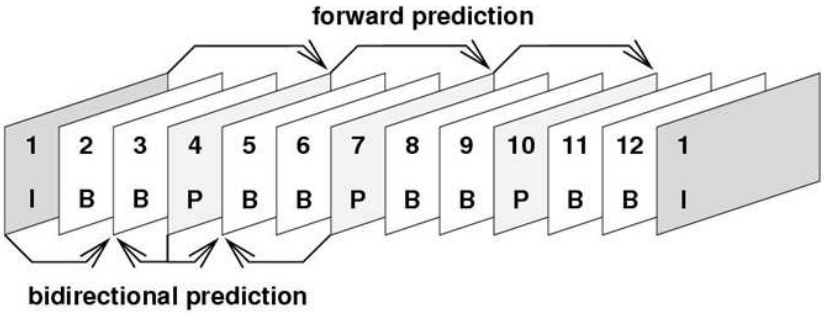


Figure 4.1: MPEG group of picture (GOP).

The different types of frames convey different levels of information and accordingly are characterized by different frame sizes. The I-frame carries the largest amount of information among other frame types and is characterized by the largest frame size. The P-frame is typically around a quarter I-frame and it is the second largest frame. The B-frame, on the other hand, uses forward and backward prediction making it the smallest frame whose size is typically half a P-frame size. Sequences of MPEG video comprise group of pictures (GOP) where each GOP comprises video frames of the three different types. GOPs usually occur in a periodic fashion in between two I-frames. The sequence of interleaved frames within one GOP is decided on in advance of the whole transmission. A GOP consists commonly of 12 frames interleaved in the following sequence: “IBBPBBPBBPBB”. Figure 4.2 presents a sequence of two successive GOPs extracted from a real video file of a soccer game. The figure demonstrates a large I-frame followed by two small B-frames and one P-frame.

MPEG-coded videos have either a constant bit-rate (CBR) or a variable bit-rate (VBR). With CBR compression, the quantization scale is modified so as to achieve a fixed rate leading to quality degradation in high-motion scenes and waste of bandwidth in low-motion scenes. With VBR compression, on the other hand, more bandwidth is allocated to scenes that are hard to compress; in consequence, clear distinction is made between successive scenes of fast-moving high-action videos. In general, VBR encoding provides superior video quality with shorter compression delays. A better channel allocation is obtained in VBR video transmission as compared to CBR video transmission. As a result, VBR encoding is more commonly used in IP networks. However, due to variable bit rate transmission, more complexity and higher challenges arise in determining the amount of capacity resources required for a successful transmission of video streams.

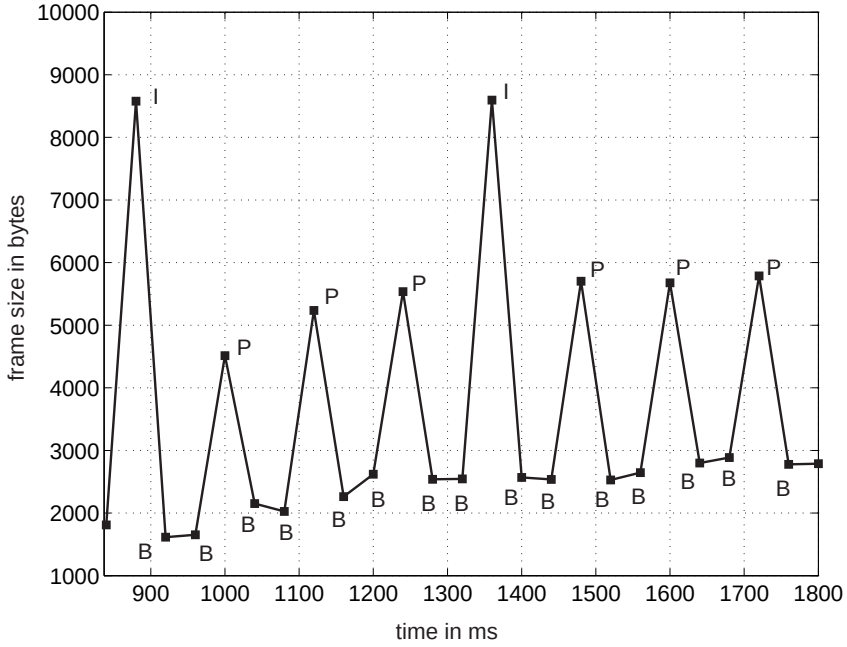


Figure 4.2: Frame sequence of a soccer video.

Without loss of generality, we assume that all video sources handled in this chapter generate MPEG-coded video traffic. However, we validate our results with MPEG-independent video models while the real videos used are MPEG-coded.

4.2 Video Sequences: Analysis and Insights

4.2.1 Theoretical Video Models

Transport of compressed video is expected to pervade computer networks in the near future. Video is commonly encoded in variable bit rate traffic to improve video quality and reduce encoding delays and yet to make efficient use of the available capacity using statistical multiplexing. Statistical multiplexing leads to variable buffering delays and losses, which negatively affect the video quality. From the perspective of a network planner, it is important to assess the impact of video traffic on the given network. To this end, various statistical source models are developed to ease the performance analysis by evaluating QoS metrics such as packet loss, delay and jitter. A survey on a number of used VBR source models is provided in [IR99]. It is not our intention to provide an exhaustive study of the available source models but only an overview of the models suitable for our purpose.

Source models used to assess MPEG-video traffic has to recreate the MPEG structure including the I-, B- and P-frames. Other characteristics like the autocorrelation function of traffic has to match those of real video sources. Source models available in the literature can be classified into two main categories, namely Markov-based models and self-similar models. The former category has an advantage of a lower computational complexity as compared to the latter, but requires many parameters (the coefficients of the Markov-chain). The autocorrelation function

of Markovian models matches pretty well with that of real video sources in the short range, the fact that makes them short range dependence models (SRD).

As to the latter category comprising the self-similar models, few parameters are required with obviously a higher complexity in generating video samples. The fractional autoregressive integrated moving-average (F-ARIMA) is an example of the self-similar models and can be used to generate traffic whose autocorrelation function can match any kind of desired autocorrelation function. Self-similar models are distinguished with their long range dependence (LRD) as opposed to Markov-based models.

Whether SRD or LRD is the most relevant for network resource dimensioning has been an ongoing debate. Efforts have been put in developing a new model that offers both SRD and LRD features [KM98]. This new model is a hybrid model sharing common properties with both Markovian models and self-similar models. It is based on the so-called $M/G/\infty$ input process that has been shown as more adequate in modeling video sources than DAR(1) (Markovian model) and F-ARIMA (self-similar model) for example. A rather comprehensive study on the issue of modeling video traffic is available in [ALS02]. In this work, we focus on the $M/G/\infty$ model and compare results with the F-ARIMA model.

4.2.1.1 F-ARIMA Model

The F-ARIMA model is developed and analysed in [GW94] [HDLK96] [LVR$^+$03]. This model requires a set of at least seven parameters to generate a trace file. The parameters are

1. the seed for the internal random number generator,
2. the Hurst parameter of the sequence to be generated (it characterizes the sequence self similarity),
3. the ratio $M_{\mathrm{X}}/M_{\mathrm{GOP}}$ where M_{X} is the average size of the frames of type X whether I-, P-, or B-frame and M_{GOP} is the average size of a GOP,
4. the type of marginal distribution of the GOP size that can be any of the following: Exponential, Beta, Gamma, Lognormal, Pareto, Uniform, and Weibull,
5. the parameters required to characterize the marginal distribution of the GOP size.

According to [KM98], the frame size of an MPEG video follows a Gamma distribution at the main region while it is better represented by a Pareto distribution at the tail region. For network dimensioning purposes, special interest lies in the tail distribution the fact that makes us consider the Pareto distribution in our work to generate video traffic traces. The F-ARIMA model generates sets of GOPs whose sizes follow the selected distribution. Within each GOP, the model applies the given ratios of item 3 in the above list to derive the individual sizes of the frames constituting each GOP. These ratios are kept constant throughout the generated distribution.

4.2.1.2 $M/G/\infty$ Model

$M/G/\infty$ has been introduced in [KM98] as a compromise between Markovian and LRD models. It is shown in the same work as the only model capable of consistently providing close

predictions to the actual queuing performance. The distinctive feature of this model is that it captures the frame size distribution of real video sequences. The main part of the frame size distribution is modeled as a Gamma distribution and the tail part as a Pareto distribution. Therefore, the required parameters are those for the Gamma and Pareto distribution parts as well as the transition point between the two distributions.

$M/G/\infty$ can be defined with the following seven parameters.

1. The seed for the internal random number generator.
2. The Hurst parameter of the sequence to be generated.
3. The mean and standard deviation of the sequence.
4. The transition point (x^*) between the Gamma and Pareto distribution parts of the video sequence.
5. The scale (a) and shape (α) parameters for the Pareto distribution part.

The overall frame size distribution of the $M/G/\infty$ model should be smooth especially at the transition points at which the nature of the distribution differs. If continuity and smoothness at the transition point are ensured, the scale and shape parameters for the Pareto distribution part are not necessary anymore.

The CCDF of a Pareto distribution is given by:

$$P_x = \begin{cases} \left(\frac{a}{x}\right)^{\alpha} & \text{if } x \geq a, \\ 1 & \text{otherwise,} \end{cases} \tag{4.1}$$

where a and α denote the scale and the shape of the Pareto distribution respectively and P_x is the probability that the frame size is greater than x.

In a logarithmic scale, the CCDF of the Pareto distribution becomes:

$$\log(P_x) = \begin{cases} \alpha\log(a) - \alpha log(x) & \text{if } x \geq a, \\ 0 & \text{otherwise.} \end{cases} \tag{4.2}$$

As for the Gamma distribution, it is more complex and can be solved numerically. The CCDF of the Gamma distribution is given by

$$P_x = 1 - \mathrm{F}_{\mathrm{Gamma}}(x), \tag{4.3}$$

where $\mathrm{F}_{\mathrm{Gamma}}$ is the cumulative distribution function of a Gamma distribution. At this point, we solve for the tangency condition at the transition point between the two distributions in order to have a smooth curve at this point.

The values of a and α can thus be derived from the following equations where T_r is the transition point and ϵ is an arbitrary small number.

$$\alpha = \frac{\log\left(\frac{1 - \mathrm{F}_{\mathrm{Gamma}}(T_r + \epsilon)}{1 - \mathrm{F}_{\mathrm{Gamma}}(T_r)}\right)}{\log\left(\frac{T_r}{T_r + \epsilon}\right)}. \tag{4.4}$$

$$a = 10^c, \tag{4.5}$$

where

$$c = \frac{\log(1 - \mathrm{F}_{\mathrm{Gamma}}(T_r)) + \alpha \log(T_r)}{\alpha}. \tag{4.6}$$

When the Gamma distribution and the transition point are known, a and α can be numerically estimated defining the necessary parameters to construct the Pareto distribution part. As a result, the tangency condition is a sufficient input to completely define the Pareto distribution.

To determine the typical range of values for each parameter, we make use of real video traces and evaluate the different parameters. Video files can then be generated using the list of required parameters whose values are randomly selected from the designated range of typical values.

4.2.2 Real Video Samples

For our investigations, we use real video traces with different properties in terms of motion, frame size, and quality. These video traces are taken from [FR00] [FR01] and they include movies, cartoons, TV sequences, and videoconferences. Each of these videos is MPEG-4 encoded in three different qualities, namely,

- low quality,
- medium quality,
- high quality.

In the simulations, we use the high quality version of the videos where a large mean bit rate range is offered starting from 0.2 Mbps to 1.1 Mbps. The original video sequences in [FR00] are 26 in number and they are 60 minutes long. We divide the original video files into 5-minute sequences and use the shortened version in the simulations. Doing so, we reduce the simulation and post-processing time as well as increase the number of available video sequences thus providing a large set of real video samples. Since using extracts from one video trace might increase the correlation among the aggregated videos especially if these extracts are rather consecutive in time, we select extracts separated by a long time interval (20–30 minutes) to reduce the correlation as much as possible.

Among the available video sequences, we extract two videos on which detailed analysis is based while we use all the available sequences for performing the simulations and validating the obtained results. The two videos selected for the particular analysis are Soccer and Teaching. Soccer is a high-motion video with a high mean bit rate and homogenous frame sizes even for I- and B-frame types. Teaching, on the other hand, is a very low-motion video with a low mean bit rate and largely varying frame sizes.

4.2.3 Comparison of Theoretical and Real Video Models

F-ARIMA and $M/G/\infty$ models are implemented and run to simulate video traces with realistic video characteristics. As an example, we configure both theoretical models to fit the Soccer video and compare the resulting traffic rate distributions. We note that the traffic rate distribution is identical in behavior to that of the frame size except for a scaling factor: one frame is deterministically generated every 40 ms.

4.2.3.1 Frame Size Distribution

The F-ARIMA model is simulated to generate video traces and tuned to match the Soccer video flow. The parameters used in configuring the F-ARIMA model are the following.

- Ratio:
  - $\frac{M_{\mathrm{I}}}{M_{\mathrm{GOP}}} = 0.126$
  - $\frac{M_{\mathrm{P}}}{M_{\mathrm{GOP}}} = 0.287$
- Hurst parameter:
  - H = 0.867
- Gamma distribution:
  - $r = 56000$ bytes
  - $\sigma = 29933$ bytes
- Pareto distribution:
  - $a = 40000$ bytes
  - $\alpha = 3.0$

As mentioned previously, the F-ARIMA model uses either of seven possible marginal distributions among which are the Gamma and Pareto distributions. All possible distributions are nonetheless tested, however, poor results are obtained in matching the resulting distribution with the real video except for the Gamma and Pareto models. In Figure 4.3, we demonstrate the fitting capability of the F-ARIMA model to the Soccer video once using the Pareto distribution and another time the Gamma distribution.

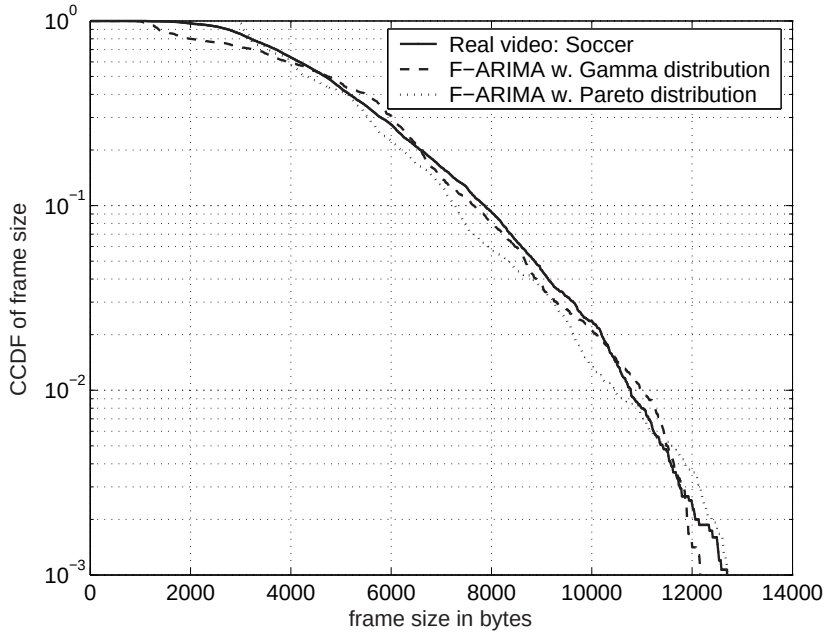


Figure 4.3: Fitting capability of F-ARIMA model to real Soccer video.

When comparing the different curves in Figure 4.3, we note that the Gamma model performs better for the main part of the distribution while the Pareto performs better for the tail part. This observation matches the conclusion of [KM98], which is applied earlier in Section 4.2.1.2. It applies on either a frame-basis or a GOP-basis due to the fact that a fixed ratio exists between the average size of one frame type (I, B, or P) within a GOP and the average size of a GOP in the F-ARIMA model. Since the tail distribution of frame sizes is dominated by the I-frames, which are the largest frames in the F-ARIMA model, there is consequently a fixed ratio between the tail of the frame-based distribution and that of the GOP-based one. In conclusion, we select the Pareto distribution as the appropriate model in configuring the F-ARIMA model due to its good matching capability to the tail distribution of real videos and this is in fact a significant property for link dimensioning purposes.

Now, we aim to match the $M/G/\infty$ model to Soccer video traffic distribution. To do so, we determine the transition point between the Gamma and Pareto parts and adapt both distributions to fit the main and the tail parts of the real video distribution respectively. In Figure 4.4, we present the CCDF obtained using the $M/G/\infty$ model and compare it to that obtained using the F-ARIMA model with Pareto distribution. We conclude that both F-ARIMA and $M/G/\infty$ are capable of modeling real video traffic; however $M/G/\infty$ results in a more accurate fit. The parameters used in determining the $M/G/\infty$ model are the following.

- Hurst parameter:
  - H = 0.867
- Gamma distribution:
  - $r = 4997$ bytes
  - $\sigma = 5377$ bytes

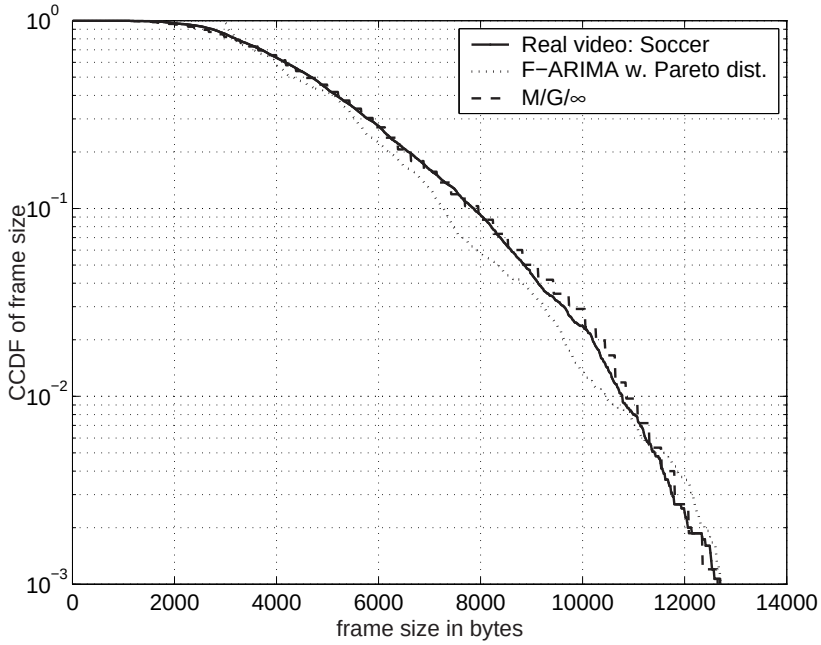


Figure 4.4: Comparison of $M/G/\infty$ and F-ARIMA models with respect to real Soccer video.

- Pareto distribution:
  - $a = 7900$ bytes
  - $\alpha = 16$
- Transition point:
  - $x^* = 10000$ bytes

4.2.3.2 Frame Size Sequence

In addition to the frame size distribution, the frame size sequence has a key role in affecting the required capacity. The frame size sequence influences in fact the frame waiting time. For example, for the same set of frame sizes, the frame waiting time differs depending on the sequence in which the frames appear on the link. Thus, to evaluate the link capacity that is capable of keeping the frame waiting time within a given threshold, the frame sequence should be analyzed.

Figure 4.5 presents a typical sequence of real video frames belonging to the Soccer video and compares it to the sequence of frames generated using F-ARIMA and $M/G/\infty$ models, which are configured to match the CCDF of Soccer frame size. The typical MPEG pattern with large I-frames followed by two small B- and P-frames is preserved in the F-ARIMA model as manifested in the figure. The sizes of different frame types of the F-ARIMA model have a fixed ratio to the GOP size and thus we can observe that all B- and P-frames inside one GOP have equal sizes. Regarding the $M/G/\infty$ model, the frame sequence is more chaotic and the typical MPEG pattern is not clear. This is due to the fact that the $M/G/\infty$ model does not differentiate between the different frame types even though it generates accurate frame size distribution.

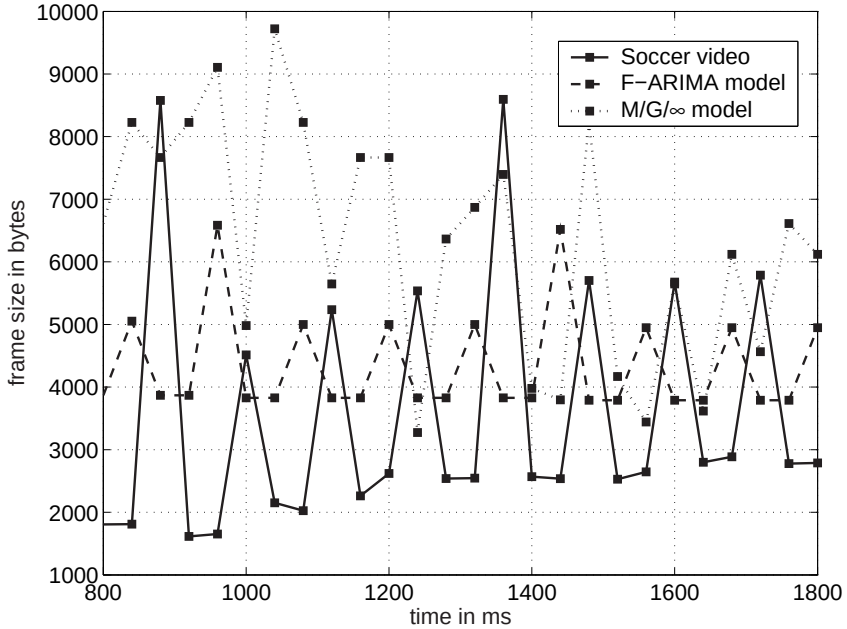


Figure 4.5: Frame size sequence comparison of $M/G/\infty$ and F-ARIMA models with real Soccer video.

In brief, while the $M/G/\infty$ model can better match the CCDF of frame size, it fails to reproduce the typical "IBBP..." sequence characterizing the MPEG format. F-ARIMA, on the other hand, provides the typical MPEG frame sequence, and hence results in a more realistic frame waiting time. As a result, it is necessary that we consider both models in our study since each captures a different behavior of real video traffic.

4.2.4 Video Traffic Modeling and Characteristics

4.2.4.1 Single Video Flows

For each video sequence, we are interested in the following traffic statistics parameters.

- Mean bit rate, r,
- Frame-based standard deviation of the traffic rate, σ_{frame},
- GOP-based standard deviation of the traffic rate, σ_{GOP},
- Frame-based 1-percentile of the traffic rate, $\gamma^{1\%}_{\text{frame}}$,
- GOP-based 1-percentile of the traffic rate, $\gamma^{1\%}_{\text{GOP}}$.

With a sufficiently large number of VBR-videos, the aggregated traffic rate comes close to a Gaussian distribution [MAS+88] as will be demonstrated later in this section. As the tail behavior of the traffic rate distribution is particularly important for link dimensioning, we consider the 1-percentile traffic rate $\gamma^{1\%}$ rather than the standard deviation for a closer match with the Gaussian distribution tail [HT03]. When the Gaussian distribution is fitted to the mean and 1-percentile traffic rate, its corresponding standard deviation is computed as follows.

$$\sigma^{1\%} = \frac{\gamma^{1\%} - r}{2.326}. \tag{4.7}$$

Figure 4.6 illustrates the computation of the 1-percentile traffic rate and the corresponding standard deviation of the Gaussian distribution. In this figure, three distributions are plotted: the traffic rate distribution of the Soccer video, the Gaussian distribution fitted to r and σ_{frame}, and the Gaussian distribution fitted to r and $\sigma^{1\%}_{\text{frame}}$. A similar way is performed on a GOP basis. We note that for a single video flow, the traffic rate is somehow normally distributed. This is due to the fact that the Soccer video is high in motion and the frame sizes are largely varying leading to a normal distribution.

In Table 4.1, we present a number of video examples with their traffic statistics. We note that the duration of each video sequence is 5 minutes.

4.2.4.2 Aggregated Video Flows

In a multiservice IP network, traffic flows sharing common characteristics and performance constraints are grouped into one traffic class and treated unanimously. Interactive video traffic forms one traffic class whose capacity requirements are to be evaluated so as to achieve premium quality level as desired.

Aggregating all active video flows into one traffic class, we can treat the total aggregated flow as a single "virtual" video flow and compute the corresponding traffic statistics. Since all video flows are assumed as MPEG-coded with 25 frame/s, one frame of each flow is expected to

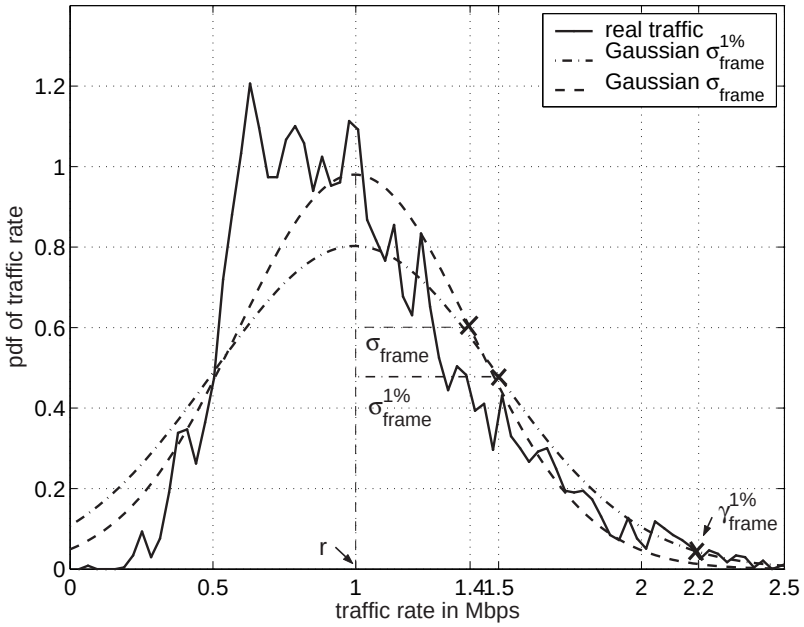


Figure 4.6: Traffic rate distribution of the real Soccer video and fitted Gaussian models: 1) using r and σ_{frame}, and 2) using r and $\sigma^{1\%}_{\text{frame}}$.

Table 4.1: Traffic characteristics of some real video flows given in Mbps.

Video trace	r	σ_{frame}	σ_{GOP}	$\gamma^{1\%}_{\text{frame}}$	$\gamma^{1\%}_{\text{GOP}}$
Soccer	0.996	0.407	0.320	2.152	1.856
Teaching	0.397	0.280	0.168	1.395	0.768
Jurassic Park	0.766	0.488	0.428	2.104	1.677
StarWars IV	0.324	0.246	0.197	1.085	0.804
M. Bean	0.530	0.354	0.164	1.959	0.917
Office	0.396	0.424	0.049	1.810	0.605
DieHardIII	0.519	0.402	0.331	2.086	1.608
The Firme	0.375	0.260	0.187	1.215	1.026
Formula 1	0.860	0.351	0.242	1.965	1.527
The Simpsons	1.206	0.465	0.337	2.541	2.121
Ski	0.910	0.529	0.440	2.461	2.097
SouthPark	0.701	0.363	0.248	1.868	1.580
ARDNews	0.762	0.556	0.428	2.221	1.982
Parkplatz	0.908	0.538	0.025	2.681	0.955
Aladdin	0.241	0.213	0.143	1.033	0.627

appear every 40 ms interval. As a result, the total frames within one 40 ms interval are grouped and considered as one "virtual" frame belonging to the "virtual" flow whose size is simply the total sizes of the associated frames. The GOPs of the "virtual" flow are identified in a similar way. The mean bit rate and the standard deviation of the "virtual" video flow process Y composed of K individual video processes X_i, $i = 1 \ldots K$, are simply given by

$$r_Y = \sum_{i=1}^{K} r_{X_i}, \tag{4.8}$$

$$\sigma_Y = \mathrm{E}[(Y - r_Y)^2], \tag{4.9}$$

where r_Y and σ_Y denote the mean bit rate and the standard deviation of the "virtual" flow process Y, and r_{X_i} denotes the mean bit rate of the flow process X_i. The different variants of the standard deviation (σ and $\sigma^{1\%}$) can be computed in a similar manner. Generally speaking, the individual video flow processes are considered as uncorrelated and thus the new standard deviation becomes

$$\sigma_Y = \sqrt{\sum_{i=1}^{K} \sigma_{X_i}^2}, \tag{4.10}$$

where σ_{X_i} is the standard deviation of X_i.

In case of any correlation among the different video flows, the resulting standard deviation of the "virtual" flow increases. In the extreme case when $Y = K \cdot X$, where $X_1 = X_2 = \ldots = X_K = X$, the new standard deviation becomes

$$\sigma_Y = K \cdot \sigma_X. \tag{4.11}$$

Figure 4.7 plots the probability density function of an aggregate of 100 different video flows and compares it to a Gaussian distribution having the same mean and standard deviation. It is illustrated that the aggregate of uncorrelated video flows tends to have a Gaussian distribution. The 100 flows are aggregated within one GOP duration in order to reduce any form of correlation resulted from the fact that all flows are MPEG-coded and thus have the same frame pattern with large I-frames and smaller B- and P-frames. In Figure 4.8, we plot the PDF of an aggregate of identical video flows while the flows are aggregated on an asynchronous basis within one GOP duration. The '$\times$' notation denotes that a number of *identical* flows are aggregated. We realize the resulting distribution to be as close to Gaussian as it was in the single flow case which indicates that no significant effect has been observed when 100 flows are aggregated as compared to the single flow case. Though the identical flows are aggregated within one full GOP, high-motion sequences that last for more than one GOP stayed to be hung together producing large "virtual" frames and similarly for low-motion sequences. Whereas, in the case of different and uncorrelated video flows, the match with the Gaussian distribution is more accurate as observed in Figure 4.7. We can conclude that the Gaussian approximation of an aggregate of video flows is quite applicable in most scenarios even when the active flows are somehow correlated. This conclusion is valid since in one single flow, the frame sizes have generally a high range of variability making the distribution close to Gaussian. As a result, the mean and standard deviation of the aggregated traffic are a proper means of characterizing the traffic.

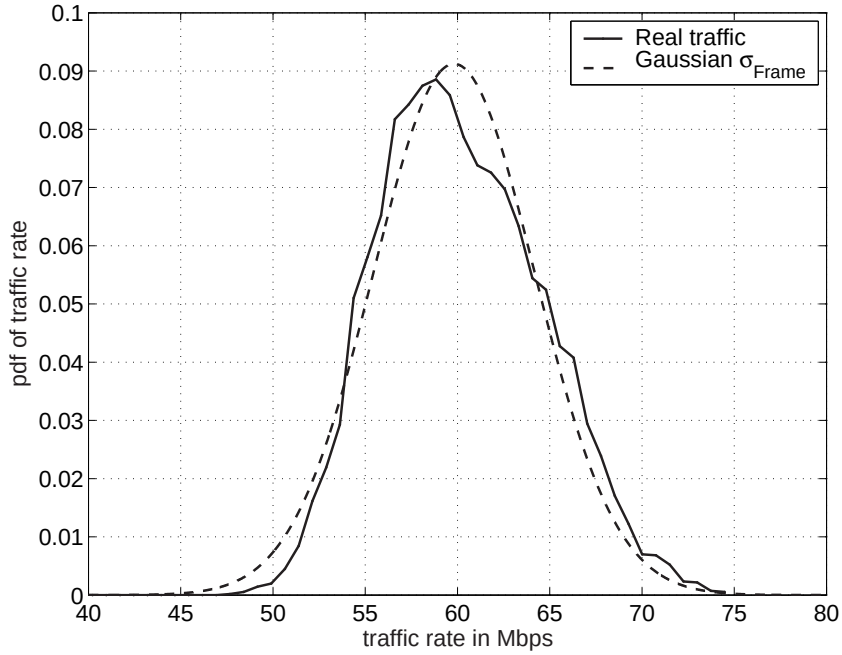


Figure 4.7: PDF of 100 real video flows vs. Gaussian traffic.

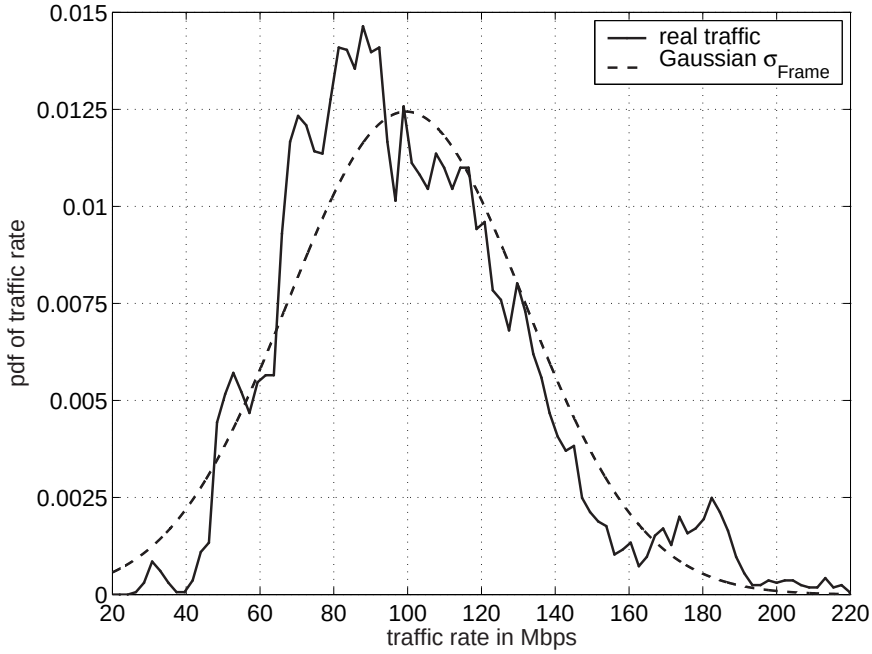


Figure 4.8: PDF of 100× Soccer videos vs. Gaussian traffic.

4.3 Realtime Conditions

The ITU-T G.114 recommendation states that the end-to-end delay of realtime applications should be limited below 150 ms in order to attain interactive communications. This delay constraint is applied to voice communications as well as video communications. In case of video communications, the end-to-end delay is constituted of different factors [BO00], namely,

- capture delay that comprises the delay of the frame grabber to record an image,
- encoding delay that comprises the compression delay of video frames,
- queuing delay that comprises the delay occurring at the different network nodes,
- propagation delay that comprises the delay dependant on the transport medium performance,
- decoding delay that comprises the decompression delay of the video frames,
- synchronization delay that comprises the delay caused by reordering of the frames and cancelation of the delay jitter,
- presentation delay that comprises the delay due to the refreshing frequency of the screen at the receiver side.

By means of different optimization and GPS synchronization (for a common time reference), it is possible to reduce the capture and presentation delays to negligible values. For realtime video transmission, it is recommended to encode (decode) a picture within one video frame period, T, i.e.

$$\mathcal{D}_{\text{encode}} + \mathcal{D}_{\text{decode}} \leq 2T. \tag{4.12}$$

A scene is normally captured at 25 frame/s, hence every 40 ms a new scene or picture has to be encoded. In order to avoid any additional delay, it is recommended that the encoding process of one picture ends in less than 40 ms duration so that the system is free when the next picture is grabbed and it can be directly encoded. The same conditions apply at the receiver side for the decoding process. As a result, the total coding delay (encoding and decoding) should remain below 80 ms. In regards to the synchronization delay, the network delay jitter is hardly predictable and it varies with the network load and structure. The maximum delay jitter is equivalent to the maximum queuing delay, which is also difficult to predict but its distribution can be estimated as done later in this chapter. The second delay component of the synchronization delay is the frame reordering delay that is dependent on the GOP structure of MPEG traffic in addition to the inter-frame coding method for B-frames [CBC99]. In Figure 4.9, we show the reordering steps performed during the coding/decoding process of an MPEG flow.

When the first picture F1 of a video flow is encoded into frame I1, no lookahead delay is required since the I-frame is independent of other frames. The next following frames are of B-type, and thus the second picture F2 and the third picture F3 are encoded in relation to I1- and P1-frames. However, at this time, P1-frame is not yet encoded and thus the encoder waits for the fourth picture F4, which is encoded into a P-frame, to appear (waiting time evaluates to $2 \cdot 40$ ms). Using I1- and P1-frames, B1-frame can now be constructed as well as B2-frame.

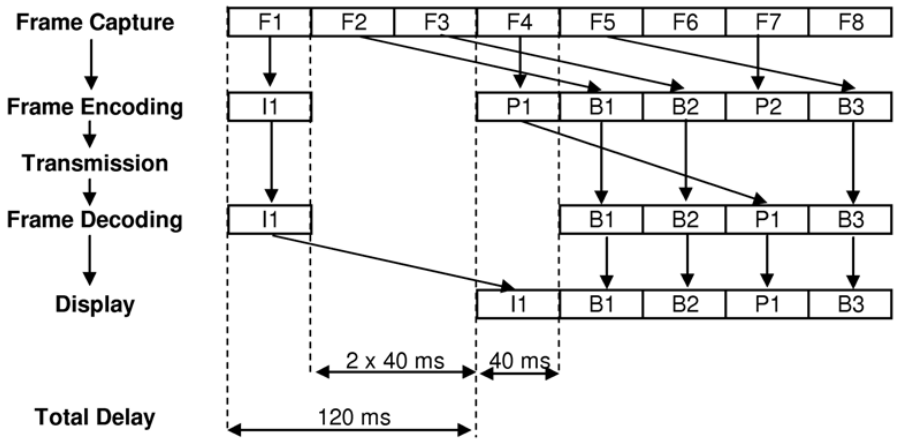


Figure 4.9: Reordering of frames of an MPEG flow during encoding/decoding process.

The encoded sequence obtained is now ”I1 P1 B1 B2“. Though this frame sequence does not correspond to the chronological order of the pictures, it will be sent as such in order to allow for immediate decoding at the receiver side. In fact, the decoder receives the I1-frame first, which is decoded directly. When the next frame (P1-frame) arrives, the receiver can readily decode it since the I1-frame is already decoded. Having I1- and P1-frames decoded, B1- and B2-frames are also directly decoded as soon as they arrive. The frames are then displayed in a chronological order which makes the P1-frame displayed after B1- and B2-frames. The maximum delay experienced in this process among the frame sequence is encountered by the B1-frame (refer to Figure 4.9).

We note that the frame sequence shown at the transmission step in Figure 4.9 is “I1 P1 B1 B2 P2”, which differs from the frame sequence of an MPEG-coded flow presented in Figure 4.1 (“I1 B1 B2 P1”). This is actually the case for only the first GOP of the video flow but it gets to the normal MPEG sequence in the following GOPs starting from the second one. The last two B-frames of the first GOP, B7 and B8, use the first I-frame of the second GOP in their encoding process. As a result, at transmission time, the I-frame of the second GOP is sent prior to B7- and B8-frames. The consequent sequence obtained is “$\mathrm{I1}_{\mathrm{GOP2}}$ $\mathrm{B7}_{\mathrm{GOP1}}$ $\mathrm{B8}_{\mathrm{GOP1}}$”. The second GOP is encoded according to the same process leading to the following sequence: “$\mathrm{I1}_{\mathrm{GOP2}}$ $\mathrm{B7}_{\mathrm{GOP1}}$ $\mathrm{B8}_{\mathrm{GOP1}}$ $\mathrm{P1}_{\mathrm{GOP2}}$ $\mathrm{B1}_{\mathrm{GOP2}}$ $\mathrm{B2}_{\mathrm{GOP2}}$ . . . ”. This sequence of frames is identical to the MPEG frame pattern presented in Figure 4.1 though some frames appearing within the duration of one GOP do not belong to the same GOP in reality.

From Figure 4.9, we conclude that the synchronization process imposes $2 \cdot 40$ ms delay at the encoder side and 40 ms delay at decoder side summing up to 120 ms. When the other delay factors like the coding, queuing and propagation delays are added, it gets hardly possible to abide by the realtime criterion (end-to-end delay less than 150 ms) if classical MPEG pattern is used. MPEG specifications, however, do also allow for a realtime adapted sequence which excludes B-frames (e.g. MPEG-4 simple profile); hence, no backward or interpolated prediction is required. As a result, no frame reordering is needed leading to negligible synchronization delay.

4.4 The Capacity Assignment Strategy

We start with a simple dimensioning model where we set the video capacity share equal to the mean bit rate. Doing so, we can serve all video traffic, however, no guarantees are given with regards to the service time that might get too long exceeding the realtime criterion. In order to reduce the service time to acceptable values, video capacity share should be increased. In this work, we consider this increase to be multiples of the standard deviation especially because the PDF of the traffic rate tends to a Gaussian distribution. Therefore,

$$C_{\text{video}} = r + m \cdot \sigma, \tag{4.13}$$

whereby C_{video} is the video capacity share, r is the mean bit rate of the video traffic whether a single flow or an aggregate of flows is considered, and m is a positive real number. σ can be either the standard deviation of the actual traffic itself or that of the Gaussian distribution fitted to the 1-percentile of the actual traffic. The traffic parameters of (4.13) can also be computed on a GOP-basis (r remains unchanged). Therefore, four variants of σ are to be considered, namely: σ_{frame}, σ_{GOP}, $\sigma^{1\%}_{\text{frame}}$, and $\sigma^{1\%}_{\text{GOP}}$. Generally, we can assume that individual video flows are independent and thus traffic statistics of the aggregate flow can be easily determined if the statistics of the individual flows are known. A similar relation coupling the required capacity with the traffic statistics has also been used in [Haß01] and the consequent works [HF02] [HT03].

Referring to (4.13), the key parameter in the dimensioning model is m. Our intention is then to determine the behavior of m and try to evaluate its range of variability. Once done, we are able to determine the capacity requirements of an aggregate flow on one network link knowing that network delay should be constrained. If done in a deterministic manner, huge capacity requirements are needed as illustrated in Chapter 2. Therefore, we need to provide premium quality with slightly softened guarantees in order to notably save resources. Our criterion in determining the value of m is then

$$\mathrm{P}\{W \geq \widehat{\mathcal{D}}\} \leq P_{\text{out}}, \tag{4.14}$$

where W is the waiting time, $\widehat{\mathcal{D}}$ is the delay threshold, and P_{out} is the outage probability that determines the frequency in which the delay threshold is exceeded.

We note that our criterion is not related to the maximum waiting time as it was for the voice service in Chapter 3. Voice traffic has commonly a constant bit rate. If VAD is activated leading to VBR traffic, its waiting time stays bounded by that of the CBR form. Having a deterministic and periodic traffic rate, the maximum waiting time of voice traffic occurs somewhat periodically and has reasonable values. In fact, video traffic having a constant bit rate can be treated in the same way as voice traffic in terms of capacity evaluation thus applying the same method presented in Chapter 3. Since video traffic is mostly encoded as VBR traffic to save huge capacity, the maximum waiting time can be very long and hardly determined. The video sequence itself as a whole is normally not available and the frame sizes cannot be predicted. Nonetheless, unlike voice traffic where the maximum waiting time occurs periodically, it is most likely that the maximum waiting time for video traffic occurs only once during the whole transmission and in consequence its value might be very large. As a result, it is rather unrealistic to account for the maximum waiting time in case of VBR video traffic transmission. In the following, we elaborate on this issue by investigating the applicability of the maximum waiting time distribution for our dimensioning problem.

4.4.1 Maximum Waiting Time Drawbacks

When video flows are aggregated, they can be grouped either synchronously or most likely asynchronously. By asynchronous, we mean that each video source is started randomly with a uniform distribution within one GOP duration. We perform a simulation in which the different cases of synchronous and asynchronous aggregation forms are studied. In each simulation run, the frame waiting times are recorded as well as the maximum frame waiting time experienced during the whole simulation. After performing large enough set of simulations, the frame waiting time and maximum frame waiting time distributions are computed. In fact, the maximum frame waiting time is equivalent to the maximum waiting time and thus these two terms can be used interchangeably. For the simulations, we use the capacity value that allows video frames to abide by the given 40 ms delay in a probability of more than $(1 - 10^{-3})$ if the video flows are synchronously aggregated.

We assume at first that all active video flows are identical and consider "10$\times$ Soccer" as an example. In Figure 4.10, we plot the different curves corresponding to the frame waiting time of the synchronous case, the maximum waiting time, and the frame waiting time of the asynchronous case.

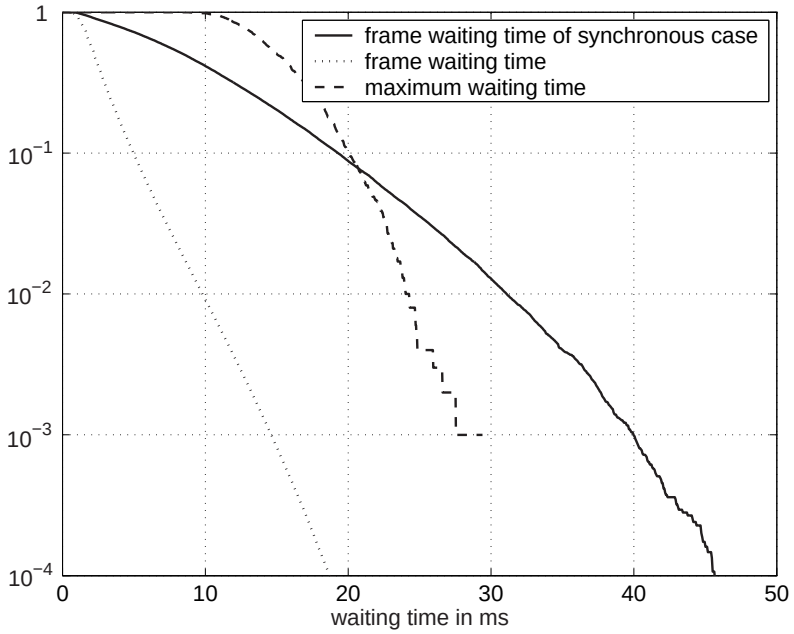


Figure 4.10: Waiting time distributions of an aggregate of 10 identical Soccer videos.

The maximum waiting time distribution in Figure 4.10 starts as an upper bound for the different distributions even for the frame waiting time of the synchronous case. The curve of the maximum waiting time crosses that of the frame waiting time at around 20 ms and falls behind it. This indicates that for low delay values, the maximum frame waiting time exceeds the given delay threshold more frequently than the frame waiting time of the synchronous aggregation. However, if a higher delay threshold is allowed, the maximum waiting time is exceeded in very low probability as compared to the frame waiting time of the synchronous case. As a result, it is a huge waste of resources if dimensioning is performed based on the synchronous case.

This conclusion is strongly validated in videos having bursty traffic such as Teaching for example which is characterized by large I-frames and tiny B- and P-frames. In the synchronous case, all I-frames are transmitted simultaneously causing a gigantic "virtual" frame that fills up the buffers and thus a very high service rate is required to keep the waiting time limited to desired values. Figure 4.11 demonstrates this concept and shows how the frame waiting time in the synchronous case largely exceeds the maximum frame waiting time for the same outage probability.

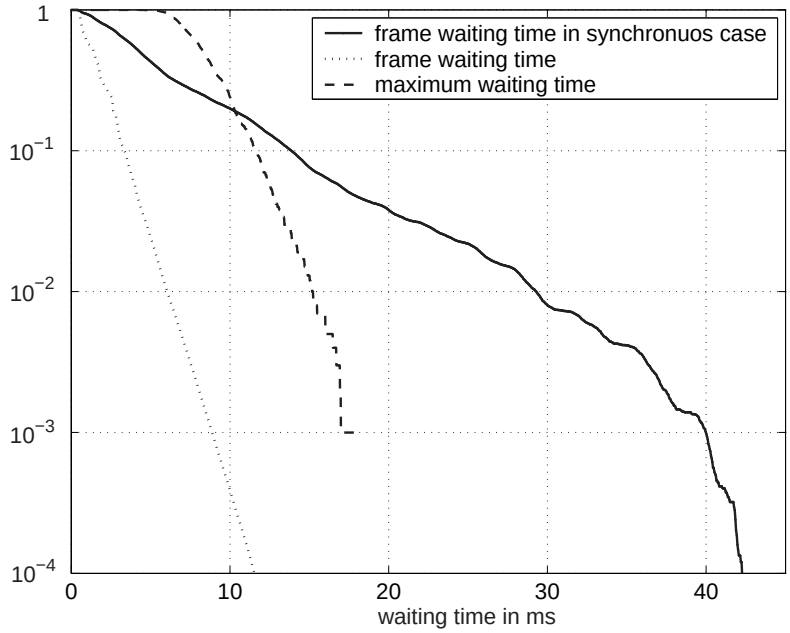


Figure 4.11: Waiting time distributions of an aggregate of 10 identical Teaching videos.

A random aggregation of a number of video flows enables the large I-frames to be compensated for by smaller B- and P-frames of other flows. This effect is less apparent for high motion videos like Soccer since frames of the different types are rather large in size. For the same quality criterion, the frame waiting time at 10^{-3} is around 15 ms for Soccer while it is only 8 ms for Teaching. In case of asynchronous aggregation of a number of different videos, we observe that the effect of random aggregation has a huge impact on saving capacity resources due to the statistical multiplexing of the flows. This observation is manifested in Figure 4.12 that plots the waiting time distributions of an aggregate of different video flows generated by the F-ARIMA and $M/G/\infty$ models (these theoretical models were shown earlier to share common characteristics to real video models). In case of F-ARIMA generated flows, we observe that a difference of 22 ms exists between the maximum waiting time and the frame waiting time of the synchronous case for an outage probability of 10^{-3} and thus even if the maximum waiting time is used for link dimensioning, huge resources can be saved as compared to the synchronous case. In the case of $M/G/\infty$-generated videos, the frame waiting time of the synchronous case is only 7 ms away from the maximum waiting time for an outage probability of 10^{-3}. This is due to the fact that $M/G/\infty$ does not generate deterministic pattern and so synchronous aggregation does not lead to having the huge frames appear together. As to the maximum

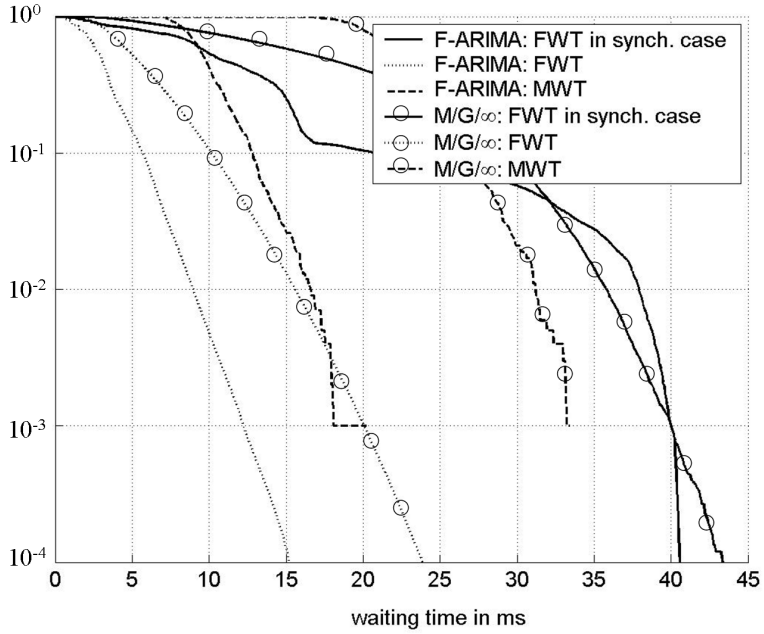


Figure 4.12: Waiting time distributions of an aggregate of 10 different video flows (FWT is frame waiting time and MWT is maximum waiting time).

waiting time distribution, it is not a practical means as well for link dimensioning for VBR-coded video. It is because this distribution can largely be affected by a single huge frame of one video flow among a number of other flows. In this case, the maximum waiting time distribution does not consider the global traffic characteristics, however, it only accounts for exceptionally large frames or single frames among a huge sequence of frames.

After setting the performance criterion, we recall the dimensioning model which still lacks a key parameter for its usage that is m. In the following we try to evaluate m for single and aggregate video flows.

4.4.2 Single Video Flow

The minimal link capacity that assures the QoS criterion of (4.14) has to be evaluated. Figure 4.13 plots the CCDF curves corresponding to $P_{\text{out}} = 10^{-3}$ and $\widehat{\mathcal{D}} = 5$ ms, 10 ms and 40 ms respectively. For each delay threshold, a different capacity is required to serve the video frames within the given threshold. For a delay constraint of 5 ms, a link capacity of at least 20.32 Mbps is needed. If higher network delays are allowed, link capacity decreases to 10.17 Mbps and 2.54 Mbps for 10 ms and 40 ms delay constraints, respectively.

For further analysis, we select $P_{\text{out}} = 10^{-3}$ and $\widehat{\mathcal{D}} = 40$ ms, and evaluate the required capacity share for serving one video flow within the given constraints. Later in this chapter, we study the impact of varying the delay threshold on the dimensioning model. Based on the computed capacity value, we determine m depending on the type of standard deviation. We have also generated random videos using F-ARIMA and $M/G/\infty$ models and computed the corresponding values of m similarly. The main results are summarized in Table 4.2.

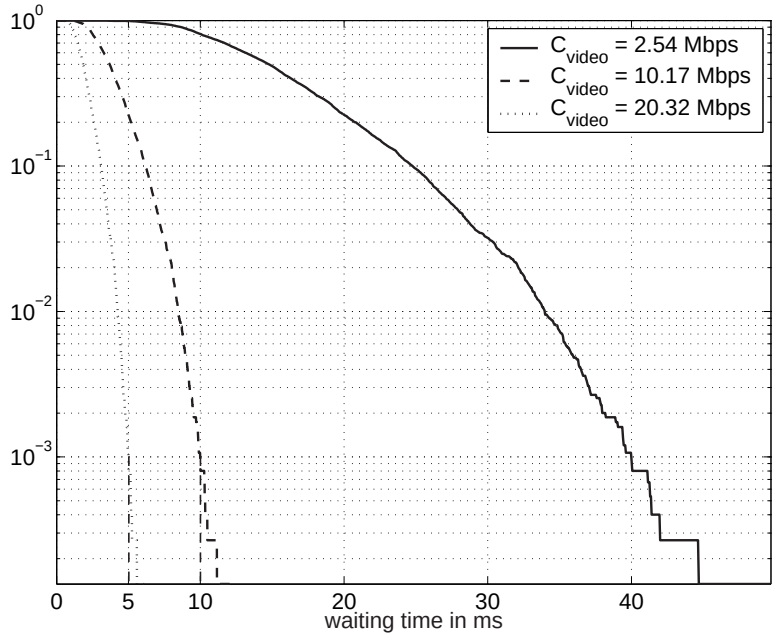


Figure 4.13: CCDF of frame waiting time of Soccer for various delay thresholds.

Table 4.2: Computed values of m depending on the four standard deviation variants of high quality MPEG-4 coded video: $P_{\text{out}} = 10^{-3}$ and $\widehat{\mathcal{D}} = 40$ ms.

Video trace	C_{video} in Mbps	$m(\sigma_{\text{frame}})$	$m(\sigma_{\text{GOP}})$	$m(\sigma^{1\%}_{\text{frame}})$	$m(\sigma^{1\%}_{\text{GOP}})$
Soccer	2.54	3.79	4.83	3.11	4.18
Teaching	1.48	3.86	6.44	2.52	6.79
Jurassic Park	2.46	3.47	3.95	2.94	4.33
StarWars IV	1.33	4.09	5.11	3.08	4.88
Office	1.84	3.41	29.37	2.38	16.11
Aladdin	1.81	7.37	10.99	4.61	9.46
Mr. Bean	2.27	4.91	10.61	2.83	10.45
Formula 1	2.47	4.58	6.65	3.39	5.62
The Simpsons	3.08	4.03	5.57	3.27	4.77
Ski	2.76	3.50	4.20	2.77	3.63
ARDNews	2.71	3.50	4.55	3.11	3.71
Parkplatz	2.70	3.33	72.11	2.35	88.45
ARD Talk	1.98	4.87	9.27	2.91	6.32
Boulevard Bio	2.35	4.70	8.60	3.10	8.63
Star Trek First Contact	1.53	4.70	5.08	3.08	3.48
Robin Hood	2.77	4.99	6.36	3.06	4.44
1 RND F-ARIMA	4.61	4.47	6.02	2.86	5.37
1 RND M/G/8	2.64	4.59	6.26	3.64	5.21

From the values of m provided in Table 4.2, we realize that m varies within a certain range depending on the standard deviation variant selected. It is apparent in this table that m is almost constant with a small range of variation when the frame-based standard deviation variants are used (σ_{frame} and $\sigma^{1\%}_{\text{frame}}$). In Table 4.3, we compute the mean and the standard deviation values of m to assess its variability behavior. If σ_{frame} is used, the standard deviation of m does not exceed 0.9 and yet if $\sigma^{1\%}_{\text{frame}}$ is used, it stays below 0.48. This infers that m is independent of the type of video and its motion level, thus represents an advantage for network planning. However, if the GOP-based standard deviations are used, m has a wide range of variation depending on the motion level of the videos. The standard deviation of m even exceeds its mean value in some cases. It grows large especially in low-motion videos like Parkplatz or Office where very little changes are expected from one scene to the other. The value of m reaches 72.11 for Parkplatz and 29.37 for Office if σ_{GOP} is used while it stays around 3 for the same videos if any of σ_{frame} or $\sigma^{1\%}_{\text{frame}}$ is used. This behavior can be explained by the fact that in low motion videos, all GOPs have almost the same size where a large I-frame is always followed by very small B- and P-frames. Having this in mind, the standard deviation of the total GOP size gets to shrink (for the low-motion Parkplatz video, $\sigma_{\text{GOP}}/r = 0.03$, while for the high-motion Soccer video, $\sigma_{\text{GOP}}/r = 0.32$). To compensate for this effect, m grows large enough in order to result in an appropriate capacity share that is capable of serving all traffic within the desired time limits. This behavior is nonexistent on the frame-based granularity as demonstrated in Table 4.2. In fact, for Parkplatz, $\sigma_{\text{frame}}/r = 0.59$ and for Soccer, $\sigma_{\text{frame}}/r = 0.40$ and thus the results using frame-based standard deviations (σ_{frame} and $\sigma^{1\%}_{\text{frame}}$) are motion-independent.

Table 4.3: Statistics related to the value of m using the different standard deviation variants.

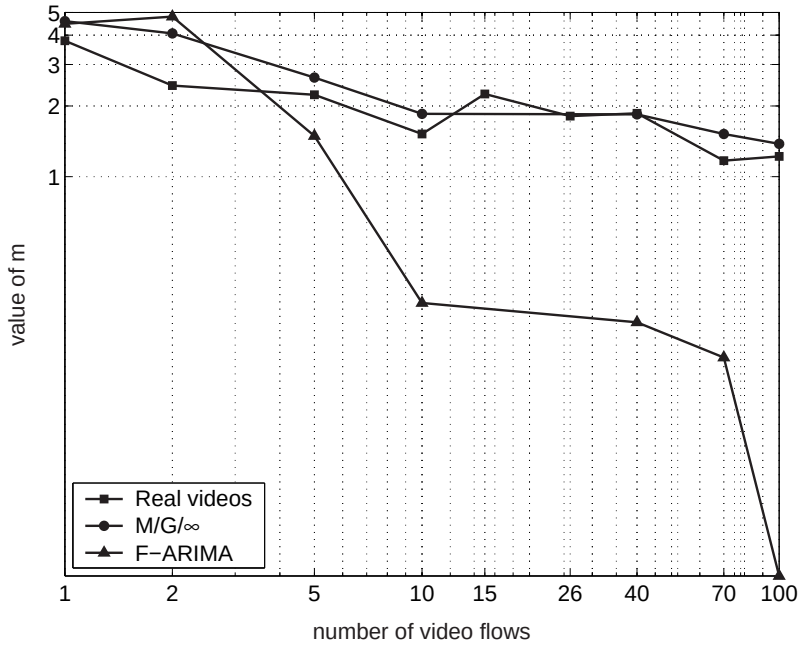
	$m(\sigma_{\text{frame}})$	$m(\sigma_{\text{GOP}})$	$m(\sigma^{1\%}_{\text{frame}})$	$m(\sigma^{1\%}_{\text{GOP}})$
Mean value of m	4.47	10.2	3.10	9.47
Standard deviation of m	0.90	13.31	0.48	16.13

4.4.3 Aggregated Video Flows

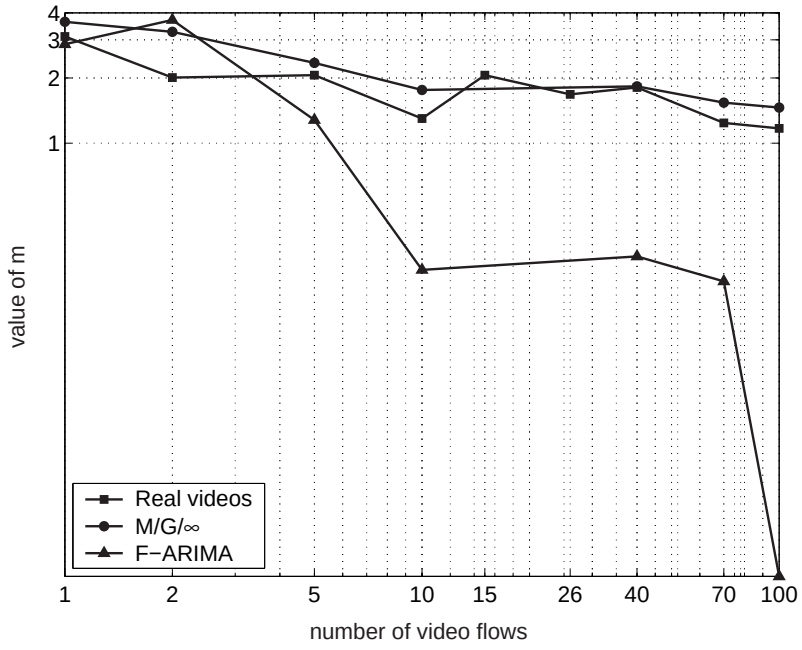
In this section, we discuss the most practical case that assumes different videos aggregated asynchronously. Other possible but unlikely cases are handled in [Unt04] such as identical flows aggregated synchronously and asynchronously and different flows aggregated synchronously.

Different videos started randomly within one GOP generate a "virtual" flow with smoother traffic. While aggregating a number of flows, statistical multiplexing comes into play and capacity is saved. In Figure 4.14a, we plot the value of m in regards to the number of videos. We show that m slightly decreases and fluctuates within a fixed interval between 1 and 2 when the number of videos exceeds 5. For a small number of aggregated flows, traffic characteristics of the final "virtual" flow is highly dependent on each of the individual flows. Thus, it is recommended to apply the results obtained for a relatively high number of videos, where the statistical multiplexing effect takes place and results get decoupled from the characteristics of individual flows. In the figure, the "Real Videos" and the "$M/G/\infty$" curves are very close highlighting the fact that the latter model is applicable for simulating real video traffic. Therefore, while planning a given network, $M/G/\infty$ model can be used in generating example flows within the network and testing the network readiness. Whereas for the case of F-ARIMA model, the corresponding curve notably falls behind the actual value. Figure 4.14b plots similar curves to the former

figure with $\sigma^{1\%}_{\text{frame}}$. We note that similar results are obtained and the same conclusions can be drawn.



(a)



(b)

Figure 4.14: Value of m with respect to the number of different video flows (a) using σ_{frame} (b) using $\sigma^{1\%}_{\text{frame}}$.

Interestingly enough, the ratio between GOP-based standard deviations and frame-based standard deviations remains almost constant when the number of flows increases. For instance, according to Table 4.4, $\sigma_{\mathrm{GOP}}/\sigma_{\mathrm{frame}}$ evaluates to 0.59 for 10 real video flows and $\sigma_{\mathrm{GOP}}/\sigma_{\mathrm{frame}}$ evaluates to 0.57 for 100 real video flows. As a result, we expect similar results to be obtained when GOP-based standard deviations are used. We tried GOP-based standard deviation variants and in fact, we obtained similar results to the case of frame-based ones.

The reasoning behind the slight decrease in m with the number of flows is that the traffic of aggregated flows gets to be smoothened. Hence, frame management and servicing at the node buffers becomes less critical. The ratio of the required capacity share to the traffic mean rate decreases slightly as manifested in Table 4.4 where $C_{\mathrm{video}}/r = 1.30$ for 10 real video flows and $C_{\mathrm{video}}/r = 1.08$ for 100 real video flows.

Concerning the results of F-ARIMA videos, high variation in m is obtained. For a large number of video files, m approaches 0 when any of the standard deviation variants is used. For example for 100 video flows, m is computed as 0.02 if σ_{frame} is used and 0.01 if $\sigma^{1\%}_{\mathrm{frame}}$ is used. Thus, the required capacity share is almost equal to the mean bit rate of the traffic (refer to (4.13)). This behavior has occurred due to the very systematic way in the F-ARIMA-created MPEG pattern that shows complementarity effect among the flows. The aggregate flow of 100 F-ARIMA videos has $\sigma_{\mathrm{frame}} = 39.62$ and $\sigma_{\mathrm{GOP}} = 0.85$. The difference between both deviations is huge, indicating that the "virtual" GOPs have almost equal sizes leading to very small standard deviation while the "virtual" frames maintain their variability among each other leading to high standard deviation.

Table 4.4: Computed values of m depending on the four standard deviation variants of a number of aggregated different video flows.

Video trace	C_{video} in Mbps	$m(\sigma_{\mathrm{frame}})$	$m(\sigma_{\mathrm{GOP}})$	$m(\sigma^{1\%}_{\mathrm{frame}})$	$m(\sigma^{1\%}_{\mathrm{GOP}})$
5 Real videos	5.63	2.23	2.96	2.06	2.82
10 Real videos	8.34	1.52	2.59	1.30	2.73
15 Real videos	13.46	2.25	3.11	2.06	2.64
26 Real videos	21.05	1.81	2.34	1.68	2.36
40 Real videos	31.10	1.86	2.67	1.81	2.66
70 Real videos	47.14	1.17	2.24	1.24	2.08
100 Real videos	65.17	1.22	2.15	1.17	2.15
2 F-ARIMA	4.59	4.80	5.59	3.71	5.06
5 F-ARIMA	7.59	1.49	13.40	1.28	18.27
10 F-ARIMA	11.32	0.29	1.73	0.26	2.30
40 F-ARIMA	47.22	0.24	1.87	0.30	1.94
70 F-ARIMA	79.13	0.17	1.36	0.23	1.81
100 F-ARIMA	89.95	0.02	0.74	0.01	1.47
2 $M/G\infty$	4.83	4.07	5.44	3.27	4.61
5 $M/G\infty$	8.29	2.64	3.43	2.35	3.22
10 $M/G\infty$	13.27	1.85	2.37	1.76	2.52
40 $M/G\infty$	50.02	1.84	2.15	1.83	2.42
70 $M/G\infty$	81.03	1.52	1.77	1.54	2.00
100 $M/G\infty$	117.66	1.38	1.55	1.46	1.94

4.4.4 Choice of the Standard Deviation Variant

In this section, we intend to investigate each of the standard deviation variants and decide upon one of them. Obviously, frame-based standard deviations provide more granular information than that of GOP-based deviations. For a low-motion video, the ratio σ_{frame}/r increases because B- and P-frames get smaller in size relative to I-frames due to the fact that low-motion videos are characterized by little changes from one scene to the other and so the difference in information between the scenes is minor leading to small B- and P-frames. At the same time, the ratio σ_{GOP}/r is almost constant yet it even decreases in some cases with respect to the number of videos. The ratio C_{video}/r increases for a low-motion sequence, due to large I-frames again that cause high I-frame rate with respect to the mean bit rate and consequently high capacity should be provided in order to absorb the resulting bursts.

In few words, for low-motion videos, C_{video}/r increases, σ_{frame}/r increases, and σ_{GOP}/r stays constant or even decreases. Consequently, $m(\sigma_{\text{frame}})$ remains unchanged while $m(\sigma_{\text{GOP}})$ increases. We consider for example Parkplatz video that represents a video sequence of a surveillance camera in a car-park, $m(\sigma_{\text{GOP}})$ is computed to be around 72 while $m(\sigma_{\text{frame}})$ keeps its value of 3. For high-motion sequences, the different frame types have comparable sizes due to the fact that the difference in information between one scene and the other is large and hence B- and P-frames encode rather lots of information causing high frame sizes. Therefore, frame-based and GOP-based standard deviations lead to similar values of m.

Based on the previous argument, we can conclude that frame-based standard deviations are more adapted for network dimensioning than GOP-based standard deviations due to their motion-independence feature especially for single flows. Table 4.2 illustrates that m maintains an almost constant value for nearly all types of videos whether σ_{frame} or $\sigma^{1\%}_{\text{frame}}$ is used.

At this point, we are interested in finding out whether σ or $\sigma^{1\%}$ are more appropriate for our purpose. In fact, for dimensioning purposes, the tail distributions of the waiting time represent our region of interest since we aim to account for most cases except for these few cases that are covered by the tail distribution. If the tail distribution was inaccurately determined then we might discard more packets than intended (allow more packets to experience high delays) thus leading to higher outage probability and vice versa. As a result, it is very important to provide accurate estimation of the tail distribution. Clearly, using σ provides better match to the main part of the traffic rate PDF than $\sigma^{1\%}$, while using $\sigma^{1\%}$ provides a better match at the tail distribution of the traffic rate PDF. Frames forming the tail distribution of the traffic rate PDF are the largest frames and, thus, they contribute to the highest frame waiting time. In the simulation results as well, we note that the variability of $m(\sigma^{1\%})$ is the least among others whether a single flow is considered or an aggregate of flows. Finally, we can decide for $\sigma^{1\%}_{\text{frame}}$ as the most appropriate standard deviation variant to be used for network dimensioning.

4.4.5 Choice of Theoretical Video Models

F-ARIMA generated videos provide a typical MPEG-pattern composed of I-, B-, and P-frames. The frame waiting time within network nodes is highly dependent on the frame sequence rather on the frame size distribution. When F-ARIMA generated videos are aggregated synchronously, meaning that all I-frames are grouped together, B-frames and so is P-frames, the frame sequence of the "virtual" flow is highly dependent on the individual sequences and thus the resulting waiting time distribution is also dependent on the individual sequences. Since $M/G/\infty$ does not

produce the MPEG-typical pattern, it performs poorly in the case of synchronous aggregation. As a result, the F-ARIMA model is the more suitable in the synchronous aggregation case though the $M/G/\infty$ generates more realistic frame size distributions which can perfectly fit with real video frame size distributions.

On the contrary, when a number of different videos are aggregated randomly within one GOP duration, the different frame types of the different videos mix together and thus the "virtual" I-frame of the resulting aggregate flow can be comprised of I-, B- and P-frames. This causes rather smoothened traffic where the initial traffic sequence of the individual flows has slight impact on the final traffic sequence of the aggregate flow. In conclusion, the $M/G/\infty$ model is the more suitable in this case. In practice, video flows are randomly aggregated and so it is advised to use the $M/G/\infty$ model unless otherwise specified (i.e. synchronous aggregation).

4.4.6 MPEG Adaptation for Realtime Transmission

The frame-type sequence of MPEG-coded videos comprising I-, B- and P-frames is not well adapted to realtime transmission due to the high coding and decoding delays required. The presence of B-frames imposes extra delay to the coding process due to the fact that B-frames make use of forward and backward prediction. For this reason, MPEG coding without B-frames is needed. In regards to $M/G/\infty$, the model generates frame sizes with a realistic distribution but no special pattern of the frames as in MPEG-coded sequences is taken into account. $M/G/\infty$ has been shown earlier to produce satisfactory results which closely match those of real videos in case of random aggregation of different flows. This implies that our results are independent of the MPEG pattern whether a B-frame is used or not though the real videos considered comprise B-frames.

In the modified pattern of MPEG coding, the coder encodes P-frames, which are typically one-quarter of an I-frame, instead of coding B-frames, which are generally one-eighth of an I-frame. However, we recall that network delay is mostly influenced by the largest frames since they cause the longest frame waiting time which represents the main interest in the context of network provisioning. As a result, the replacement of B-frames with P-frames helps in notably reducing the coding delay and has very slight effect on the frame waiting time. This makes our results still applicable to the case of MPEG coding without B-frames.

To verify this conclusion related to the applicability of our results to the modified MPEG coding, we carry out new simulations. Among the available real videos, none is coded without B-frames. So, we use the F-ARIMA model to generate sequences without B-frames, send them to a network node, and record the frame waiting time when the capacity share allocated to video traffic is set according to our dimensioning model. The value of m is set to the value found out earlier (i.e. $m = 3$), the outage probability to 10^{-3}, and the delay threshold to 40 ms assuming that the flow crosses one network node.

In Table 4.5, we present the results obtained and check whether the QoS criterion is met. In the given examples, the provided link capacity was enough to serve the available traffic within the given 40 ms delay limit. Although for the case of different flows aggregated randomly, the $M/G/\infty$ model generates closer results to actual flows than the F-ARIMA model with modified MPEG coding, these examples give us insights about the actual performance of real videos.

Table 4.5: MPEG-coded videos without B-frames tested on a network link dimensioned according to (4.13): $m = 3$, $\widehat{\mathcal{D}} = 40$ ms, and $P_{\text{out}} = 10^{-3}$.

Video trace	m	$\sigma^{1\%}_{\text{frame}}$	$m(\sigma^{1\%}_{\text{frame}})$	C_{video}	P $\{W \geq 40$ ms$\}$ $\leq 10^{-3}$?
2× F-ARIMA	1.508	0.853	3	4.07	$\surd$
10× F-ARIMA	7.451	3.297	3	17.34	$\surd$
100× F-ARIMA	42.058	3.406	3	52.27	$\surd$
2 F-ARIMA	1.583	0.961	3	4.47	$\surd$
10 F-ARIMA	11.778	1.479	2	14.74	$\surd$
100 F-ARIMA	107.39	3.985	2	115.36	$\surd$

4.4.7 Video Coding Quality Level

Videos can be coded with different quality levels. In order to study whether the quality level affects the dimensioning model, we tested the model with MPEG-coded video having low, medium, and high quality levels. It is noted in [FR01] that the variability of the frame sizes increases if the encoding quality decreases. In fact, for lower quality level, a higher compression ratio can be obtained for most of the frames if some details of the picture are ignored and not encoded. However, the changes from one scene to the other should still be encoded. This makes traffic rate more variable at the frame level within one GOP. Consequently, the ratio of the frame-based standard deviation to the mean bit rate increases. GOP size variability decreases since all GOPs have almost equal sizes making GOP-based standard deviations less influenced by the coding quality level. Table 4.6 presents the different statistics parameters for an example video, Aladdin. If high quality coding is used in this example, $\sigma_{\text{frame}}/r = 0.88$ and if low quality coding is used, $\sigma_{\text{frame}}/r = 1.39$ (the ratio increases for lower quality). On the contrary, $\sigma_{\text{GOP}}/r = 0.59$ for high quality videos and 0.54 for low quality videos. Hence, the coding quality level has a slight influence on σ_{GOP}/r ratio.

As to the ratio of C_{video} to the mean bit rate, it increases so as to abide by the QoS criterion in case of bursty traffic which is resulted due to the increased variability in frame sizes for reduced quality level. For example, $C_{\text{video}}/r = 7.5$ for high-quality coding, 9.6 for medium-quality coding, and 10.2 for low-quality coding. If C_{video}/r and σ_{frame}/r increase for lower quality coding, m then remains almost constant if frame-based deviations are used and particularly if $\sigma^{1\%}_{\text{frame}}$ is used where m evaluates to an approximate value of 4 as demonstrated in Table 4.7. As mentioned earlier, σ_{GOP}/r remains unchanged causing m to increase in order to compensate for the increase in the required capacity share. If GOP-based standard deviations are used, m has a wide range of variability. This is shown in Table 4.7 where $m(\sigma_{\text{GOP}})$ varies from 9.91 to 22.99 and $m(\sigma^{1\%}_{\text{GOP}})$ from 7.12 to 17.20.

From Table 4.7, we realize that $m(\sigma^{1\%}_{\text{frame}})$ is the most appropriate variant as it is the least one affected by the quality level. This independence of the quality level employed is particularly important for network planning. As a result, link dimensioning can be performed regardless of the quality level in which the transmitted video traffic is encoded. This also adds to the credibility of $\sigma^{1\%}_{\text{frame}}$ among the other variants.

Table 4.6: Traffic characteristics of Aladdin video with different quality levels.

Aladdin video	r	σ_{frame}	σ_{GOP}	$\gamma_{\text{frame}}^{1\%}$	$\gamma_{\text{GOP}}^{1\%}$
Low quality	0.059	0.082	0.032	0.429	0.159
Medium quality	0.083	0.090	0.052	0.438	0.251
High quality	0.241	0.213	0.143	1.033	0.627

Table 4.7: Computed values of C_{video} and m for Aladdin video with different quality levels: $P_{\text{out}} = 10^{-3}$ and $\widehat{\mathcal{D}} = 40$ ms.

Aladdin video	C_{video}	$m(\sigma_{\text{frame}})$	$m(\sigma_{\text{GOP}})$	$m(\sigma_{\text{frame}}^{1\%})$	$m(\sigma_{\text{GOP}}^{1\%})$
Low quality	0.60	9.02	22.99	4.64	17.20
Medium quality	0.79	5.73	9.91	3.38	7.12
High quality	1.81	7.37	10.99	4.60	9.43
Average of m	–	7.37	14.63	4.21	11.25
σ of m	–	1.34	5.92	0.58	4.31

4.4.8 Influence of the Delay Threshold

So far, a delay threshold of 40 ms is required for an outage probability of 10^{-3}. In this section, we investigate the impact of the delay threshold on the obtained results. In some practical scenarios, 40 ms per-hop delay for realtime services might get above the allowed end-to-end delay limit. A realtime flow is expected to cross a number of network nodes to reach its destination where the queuing delay of each node is unpredictable due to the existence of other cross traffic as one cause. Network delay increases with the number of traversed hops and network load. In this section, we test our results for the same outage probability of 10^{-3} but with reduced delay threshold values such as 20 ms, 10 ms and 5 ms so they can apply in a multihop network. These delay thresholds are assumed on a per-hop basis. Figure 4.15 and Figure 4.16 plot the values of m obtained with respect to the number of video flows for different per-hop delay thresholds. The former figure assumes identical video flows while the latter assumes different video flows. For the results, we use $m(\sigma_{\text{frame}}^{1\%})$ as recommended earlier.

Similar trend is observed in both figures where the four curves corresponding to the different delay thresholds tend to converge as the number of aggregated flows increases. In random aggregation of flows, the frames belonging to the different videos are distributed within one GOP and it is unlikely that large frames get together. Hence, buffers are unlikely to be overloaded all through the transmission time since "virtual" frames have moderate sizes. As a result, for a large number of flows, the delay threshold has no impact on the value of m.

When the required delay threshold is reduced, the link capacity is increased. This imposes a decrease in the total size of the frames present in the buffer at one time. A slight increase in the link capacity is needed to assure half of the initial delay limit; this causes the value of m to converge to a common value for the different thresholds. In Figure 4.15, the four curves of the given delay limits seem to converge to the value of 3. This occurs at the point when the total number of identical videos reach 100.

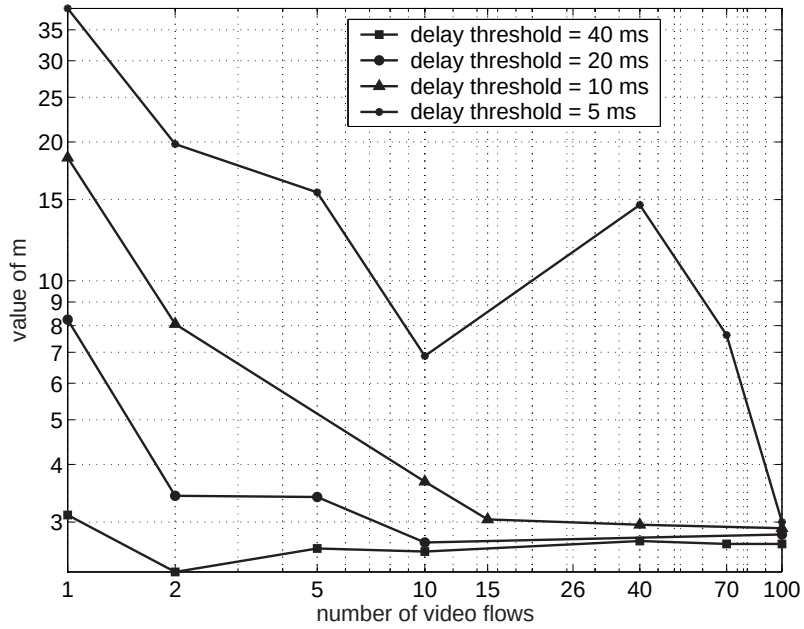


Figure 4.15: Value of m vs. the number of identical Soccer videos: $\widehat{\mathcal{D}} = [40\ 20\ 10\ 5]$ ms and $P_{\text{out}} = 10^{-3}$.

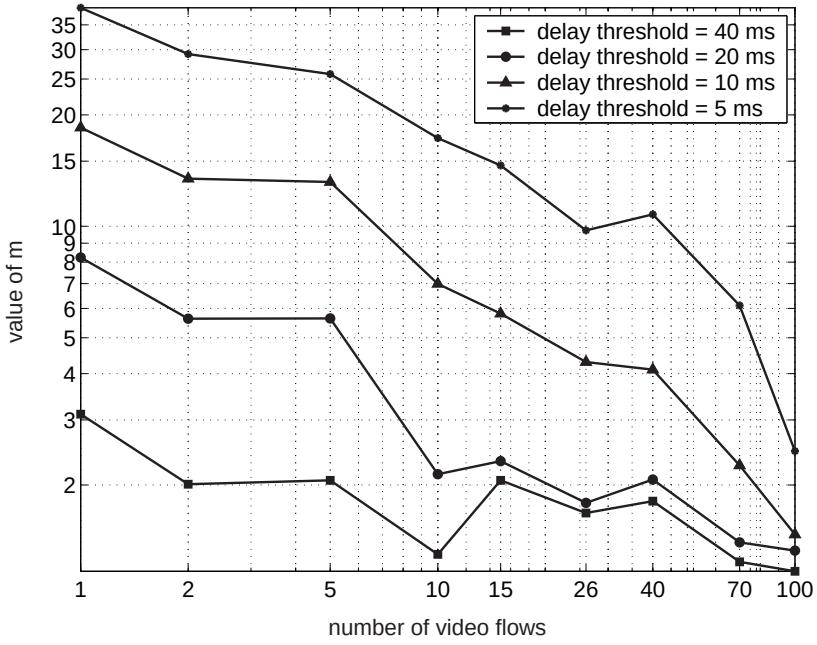


Figure 4.16: Value of m vs. the number of different real videos: $\widehat{\mathcal{D}} = [40\ 20\ 10\ 5]$ ms and $P_{\text{out}} = 10^{-3}$.

Table 4.8: Effect of changing the delay threshold on C_{video} and the value of m.

Video trace	$\widehat{\mathcal{D}} = 40$ ms		$\widehat{\mathcal{D}} = 20$ ms		$\widehat{\mathcal{D}} = 10$ ms		$\widehat{\mathcal{D}} = 5$ ms	
	C_{video}	m	C_{video}	m	C_{video}	m	C_{video}	m
2x Soccer	4.16	2.34	5.17	3.42	10.17	8.81	20.35	19.79
5x Soccer	10.30	2.63	11.87	3.40	21.31	8.06	36.47	15.55
10x Soccer	20.46	2.59	20.99	2.71	24.83	3.67	37.79	6.87
100x Soccer	202.54	2.69	207.57	2.82	210.98	2.91	214.52	3.00
Teaching +Soccer	2.66	2.01	4.94	5.63	9.87	13.45	19.75	29.19
5 Real Videos	5.63	2.06	8.99	5.64	16.07	13.18	27.91	25.79
10 Real Videos	8.34	1.30	9.57	2.14	16.70	6.99	31.89	17.31
15 Real Videos	13.46	2.06	13.92	2.32	19.96	5.81	35.20	14.60
26 Real Videos	21.05	1.68	21.29	1.79	26.91	4.30	39.10	9.75
40 Real Videos	31.10	1.81	31.74	2.07	36.65	4.10	52.76	10.77
70 Real Videos	47.14	1.24	47.69	1.40	50.61	2.26	63.76	6.11
100 Real Videos	65.17	1.17	65.88	1.33	66.55	1.47	71.10	2.47

As to Figure 4.16, a similar behavior is realized; however, the curves do not converge completely at 100 flows. When the number of aggregated videos is relatively small, the large I-frames of the videos have a significant influence on the tail of the frame waiting time distribution. This is especially apparent for a single flow case. Even for the two-flow case, we note that the capacity share is doubled if the desired delay threshold is halved. In Table 4.8, "Soccer + Teaching" aggregate flow requires 2.66 Mbps, 4.98 Mbps, 9.87 Mbps and 19.75 Mbps if the delay threshold is set to 40 ms, 20 ms, 10 ms and 5 ms respectively. This shows that the capacity share is nearly doubled when the threshold is halved. For identical flows aggregation (e.g. $2\times$ Soccer), the same effect is observed when the delay threshold is set to 20 ms, 10 ms and 5 ms where the capacity share is given by 5.17 Mbps, 10.17 Mbps and 20.35 Mbps. Capacity doubling is not apparent for $2\times$ Soccer when the delay threshold is reduced from 40 ms to 20 ms; this can be explained by the fact that the 40 ms threshold is already long enough and it is equal to the inter-frame duration of a single video, hence this allows two frames only to meet in the buffer. The waiting time can then be halved but the required capacity need not be doubled to serve the traffic within the given limit.

4.5 Summary

In this chapter, we investigated a new dimensioning model for interactive video service using real video files. The model allows for statistical QoS guarantees by defining an outage probability which determines the percentage of frames that are allowed to exceed the desired delay threshold without affecting the desired quality level. We have shown that for VBR videos, unlike CBR video and voice transmission, it is large waste of resources to account for the maximum waiting time in the capacity assignment process so we rather use the frame waiting time. Though the available video files used in this chapter are MPEG-coded, we showed that the obtained results are directly applicable to general forms of video traffic. This is done by using the $M/G/\infty$ model to generate general video patterns and comparing its results to those obtained

using MPEG-coded videos. A close match is resulted. We have also studied the manner in which the requested delay threshold affects the capacity share needed. Nonetheless, we have shown that the change in the quality level of the coded videos has a slight impact on the achieved results.

5

Robust Network Dimensioning with Uncertain Demands

In previous chapters, capacity assignment for voice and video traffic classes is performed based on a given and fixed traffic demand. In an IP network, however, the actual traffic may easily deviate from the given values due to various factors including the change in user behavior and the lack of resource control inside the network. The computed capacity values might then be inadequate for serving the available traffic with desired QoS. This problem is actually a significant hinderance facing the successful launching of realtime services over IP networks. In this chapter, we motivate this problem that is caused by traffic deviation and propose an efficient way to deal with such deviation and still achieve a highly robust network.

We focus on an abstract classification of most of today's call admission control (CAC) schemes and show how traffic deviation can occur in various levels due to the tradeoffs offered by each scheme. The more control the scheme provides, the more complex it is. In this chapter, we select the most common CAC scheme having the least complexity and apply a new approach for reducing the impact of traffic deviation on the service quality. This is achieved by means of a capacity margin that we introduce to be added to the planned link capacities. This margin is designed to account for additional traffic routed on one link when traffic deviation occurs.

The structure of this chapter is as follows. In Section 5.1, we start with some background information including a summary of the available techniques used for traffic demand estimation and an overview of call admission control mechanisms that can be used for resource control in IP networks. In Section 5.2, we elaborate on the problem of traffic deviation and highlight its drawbacks, after which we present the network model upon which our proposed solution is applied. In Section 5.3, we describe possible approaches for IP network provisioning with given traffic demand and propose a new statistical technique that offers a tradeoff solution in terms of reduced network resources and elevated service quality. For the new statistical technique, we study the distribution of traffic directed from one source to any destination, and compute a capacity margin that accounts for traffic deviation. In Section 5.4, we compare the different network provisioning approaches by means of demonstrative examples.

5.1 Overview

5.1.1 Network Tomography

One fundamental prerequisite for the traditional network planning process is demand forecasting. It actually drives the different planning stages including network design, routing, and provisioning. Demand forecasting presents the expected traffic demand which can conceptually be represented by a traffic matrix $\mathcal{A}$ that defines the amount of data transmitted between all possible origin-destination pairs. The knowledge inferred from the traffic matrix is very valuable to various traffic engineering tasks. This information about traffic load is crucial to efficiently engineer any IP network.

In [Var96], network tomography is introduced to refer to the problem of traffic matrix estimation based on aggregated byte counts measured on links. In [FGL+01], a measurement methodology is presented that derives traffic demands in an IP backbone. This study combines flow-level measurements collected at all ingress links with reachability information about all egress links. However, deriving accurate and reliable traffic matrices by means of massive measurements solely is typically very costly and usually not feasible. For this reason, statistical inference techniques are proposed in the literature as a more realistic yet accurate alternative to the direct measurement approaches. There exist several of these statistical techniques where estimation is based on varying assumptions of traffic distribution.

In one common approach that uses Bayesian inference techniques [TW98], source-destination demands are assumed to follow a Poisson distribution. This approach has been handled by [VG02] that generalizes the estimation process to account for successive measurements. The estimation in [VG02] is done in an iterative manner, which is in some way similar to the turbo decoding process [BGT93]. Another approach is based on expectation maximization algorithm to compute maximum likelihood estimates [CDWB00] and it assumes that source-destination demand pairs are modeled according to a Gaussian distribution. This study develops a time-varying statistical model to estimate an evolving traffic matrix over time using link byte counts measured at router interfaces under a fixed routing scheme. This model however does not scale to large networks. Another study of the authors in [CWYZ00] proposes a computationally scalable method that produces close estimates to results obtained by measurements. Other approaches to solve the traffic estimation problem are non-statistical like the one based on a straightforward application of linear programming where the problem is formulated into a basic optimization problem solved by standard techniques [Gol00]. Information theory has also been utilized in [ZRLD03] to estimate traffic matrices using entropy penalization. A comparative analysis of the main existing techniques is available in [MTS+02]. Another line of research proposes another means of traffic characterization where traffic matrices are replaced by a set of linear constraints representing a complex polyhedral traffic ensemble [PV03a] [PV03b].

Despite the ongoing research on traffic matrix estimation, the traffic matrix stays to be an error-prone roughly-known forecast and its reliability is insufficient for network provisioning in many practical scenarios. This fact is more critical for realtime services, which have strict QoS requirements. Complete and accurate information about possible traffic load remains to be an important issue for network planning. The author in [Mar03] assumes that the expected traffic demand is an unknown mixture of several known scenarios where each scenario is assigned an unknown weight. When traffic demands vary by busy hour for example, the traffic matrix can be directly formulated into a mixture of different busy hours. This framework assumes that the

likelihood of each scenario is known and then it attempts to optimize the average performance. If busy hour scenarios are only considered, the average "worst-case" performance is optimized rather than the average performance.

5.1.2 Call Admission Control Schemes

The concept of admission control has existed since many years. It was applied to telecommunication systems in traditional PSTN and ATM networks where a call is not permitted into the network if any of the links along its possible communication path is not capable of serving a new call with premium and stable quality all through its activity period. This type of admission control is called link admission control as it is performed on a link by link basis. This phenomenon is the main feature that distinguishes connection-oriented networks such as (virtual) circuit-switched networks from connectionless networks such as IP networks. IP networks were originally designed for best-effort services that do not require any admission control. By the evolution of realtime services in IP networks, a number of QoS mechanisms are required to achieve high quality level for interactive communications. One main QoS mechanism is admission and resource control on a per-flow or call basis that is named call admission control (CAC). Many CAC schemes have been introduced over the past years aiming at high control level within the IP network with acceptable complexity level. In IP networks, admission control is performed at dedicated locations only, e.g. at the network borders, without contacting other elements in the network even if the call utilizes part of their resources.

In [MKM04], four basic CAC approaches are introduced, each having different complexity level. These approaches summarize most of the resource management schemes available today. In this same work and other related work of the authors [MMK04], the impact of different factors such as routing, traffic distribution, and network topology on the performance of the CAC schemes has been studied. A more general study on the available CAC schemes is provided in [RBF03] where seven generic schemes are defined and investigated in terms of capacity requirements for realtime services. For each scheme, an abstract model is introduced and possible realizations of this model are presented. The CAC schemes in [RBF03] are differentiated into three categories: CAC at ingress nodes, CAC at ingress and egress nodes, and CAC at all nodes. The terms ingress and egress are adopted from MPLS terminology. Ingress refers to a border node of a network through which traffic enters the network while egress refers to a border node through which traffic leaves the network. For each scheme, an optimized dimensioning procedure is proposed in which strict QoS is required for all accepted calls. Finally, a tradeoff between CAC complexity and capacity needs is demonstrated on several network scenarios.

In the following, a brief overview about the possible CAC schemes in [RBF03] is presented.

a CAC per Ingress Link (*I-CAC*). This is probably the simplest form of CAC in which flows are admitted at the ingress node irrespective of the egress node as long as the available capacity at the ingress link is sufficient to serve all flows in a premium manner (see Figure 5.1a). This form of CAC can be realized in DiffServ networks where resource control is performed at the ingress nodes only.

b CAC per Router Interface (*II-CAC*). This is an interface-specific CAC. The admission decision is taken based on the status of the outgoing interface of each ingress router if capable of carrying a new flow or not. This form of CAC provides higher control to the network (see Figure 5.1b). It can be realized in networks employing RSVP between the ingress and egress nodes while it is only activated at the ingress.

c CAC at Ingress per Destination (*ID-CAC*). In this scheme, CAC is performed based on the destination or egress node of the call whether it can support an additional call or not (see Figure 5.1c). Possible realization of this scheme is the pipe model where end-to-end virtual connections are established for every ingress-egress pair.

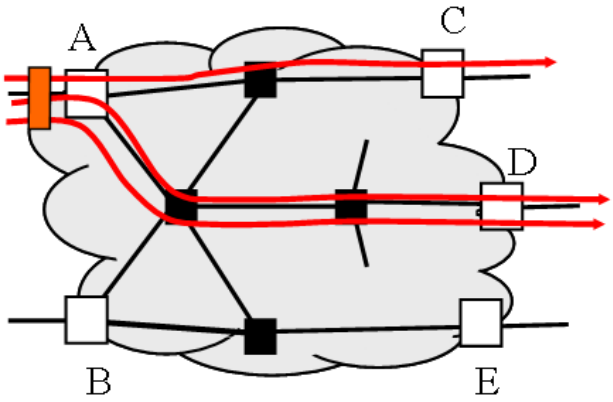
d CAC per Ingress Link and Egress Link (*I/E-CAC*). This form of admission control is an extension of *I-CAC*. In addition to resources checked in *I-CAC*, the egress link is also checked for the admission decision (see Figure 5.2a). Possible realization of this scheme is the European IST project AQUILA [Aqu00]. Whenever a call is to be set up, a signal is sent to the control agent placed at the network edge. This agent in turn contacts the other agent placed at the other network associated with the egress link. The call is established only if both agents accept it.

e CAC per Ingress Router Interface and Egress Link (*II/E-CAC*). *II-CAC* is extended by additionally accounting for the egress link of the call (see Figure 5.2b). This form is realized through an overlay IntServ concept. Ingress and egress nodes of the network domain are RSVP-enabled routers enabling an IP tunnel connecting both nodes.

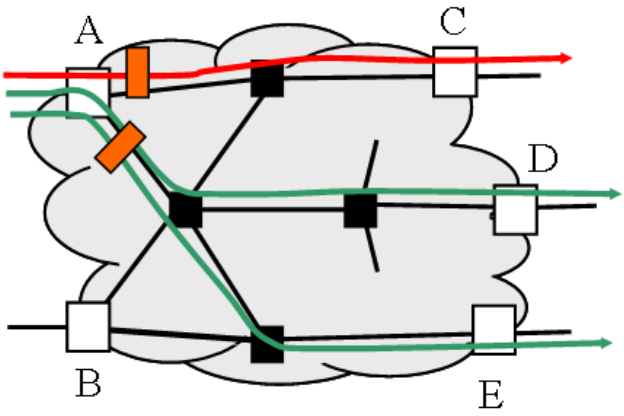
f CAC at Ingress per Destination and Egress Link (*ID/E-CAC*). In analogy to the preceding CAC schemes coordinating with egress nodes, *ID/E-CAC* is the extension of *ID-CAC* (see Figure 5.2c). Possible realization of this scheme is rather complex and no extra advantage is obtained over *ID-CAC*. However, it can still be realized by the pipe model where admission control is additionally carried out at the egress nodes.

g CAC at All Nodes (*All-CAC*). This CAC form is the most complex among all others where resource control is performed as traditionally done in circuit-switched networks. *All-CAC*, thus, achieves the highest control level among the different schemes. A call is admitted only if all links along the communication path from ingress to egress have enough capacity to accommodate a new call (see Figure 5.3).

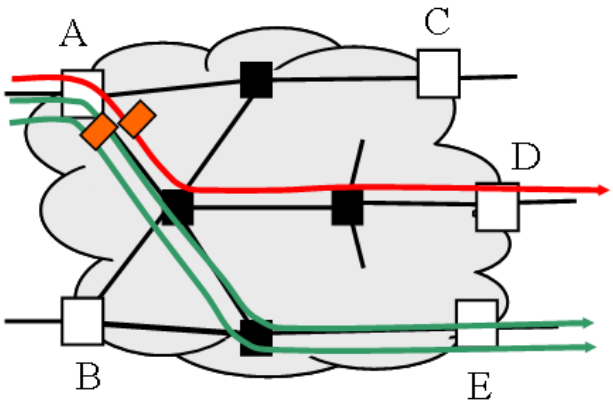
In most practical scenarios, the mean value of the offered traffic demand in the busy hour is available. Based on the given mean traffic demand and a target call blocking probability, traditional telecommunication networks such as PSTN and ATM are dimensioned according to *Erlang* formulae that take into account traffic variations around the mean and output the number of lines (trunks) required to serve the offered traffic volume. Even if the given demands are not accurately estimated, such kind of (virtual) circuit-switched networks are still able to maintain a consistent and premium quality for the ongoing calls once they are established with no phases of interruption or quality degradation. This is achieved by means of *All-CAC*, which is a tight CAC mechanism applied on a link basis. If the available resources on any link of the routing path are insufficient to accommodate a new connection, the call is simply blocked. Otherwise, once established, it is guaranteed a fixed transmission rate from source to destination all through its activity period. As a result, the actual traffic volume is controlled to conform with the given traffic demand used in network planning. Inaccurate demand forecasting might lead in this case to a higher call blocking probability, but anyway a premium service is guaranteed to all admitted active calls. Such a conclusion is not applicable however in IP-based networks employing any of the previously-stated CAC schemes, excluding *All-CAC*. In a loosely controlled network, inaccurate demand forecasting is likely to cause severe quality deterioration to the ongoing calls.



(a) *I-CAC*.

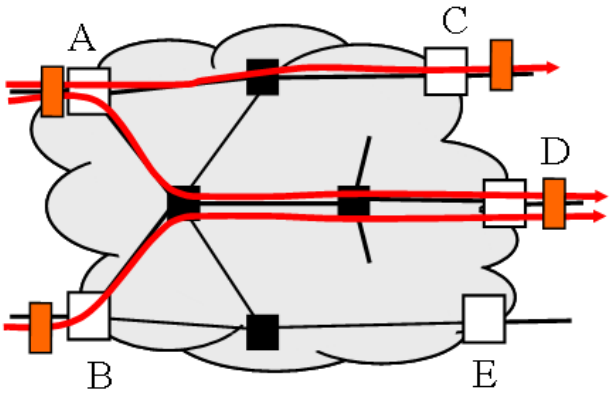


(b) *II-CAC*.

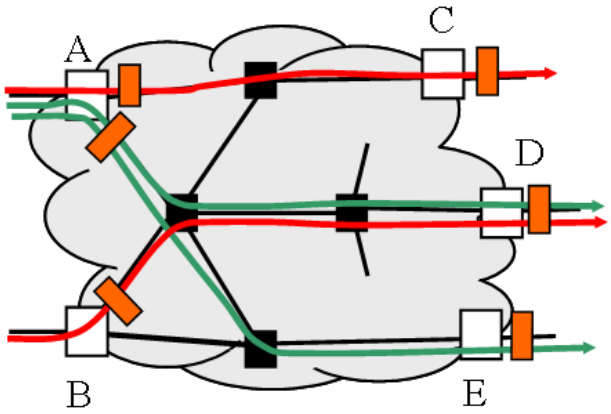


(c) *ID-CAC*.

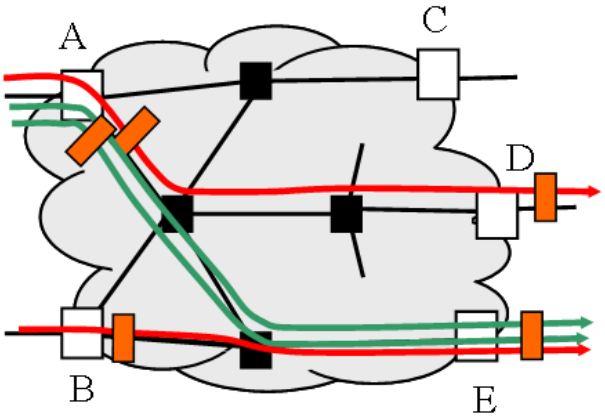
Figure 5.1: Call admission control schemes at ingress. White boxes refer to ingress nodes, black boxes to internal nodes and gray boxes to CAC units.



(a) *I/E-CAC*.



(b) *II/E-CAC*.



(c) *ID/E-CAC*.

Figure 5.2: Call admission control schemes at ingress in coordination with egress.

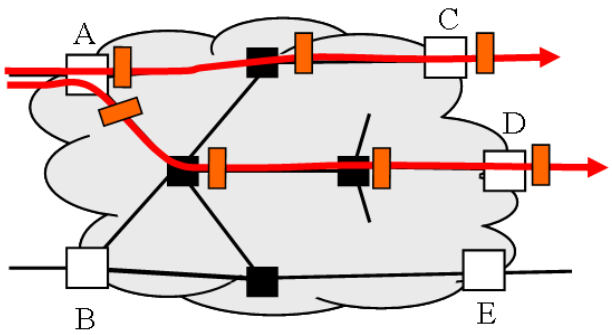


Figure 5.3: Call admission control scheme at all interfaces:*All-CAC*.

In this work, we focus on the presented concerns in integrated IP networks for appropriate network provisioning for realtime applications such as telephony and videoconferencing. We accordingly handle this issue bearing in mind the fact of having large-scale loosely-controlled IP networks. In order to protect the integrity of stream traffic generated by realtime applications, we aim at proper network planning by means of a capacity margin that accounts for traffic demand deviation inside the network.

5.2 Network Model and Problem Description

Improved quality for stream traffic is anticipated in IP networks the more they integrate circuit-switched methodologies such as *All-CAC*. Implementing such a scheme in IP networks is very complex. Less tight resource control mechanisms as the ones presented earlier are possible, varying in complexity of implementation and in the level of yielded quality. In this work, we assume the *I-CAC* scheme, which is the most practical scenario. In this network scenario, the maximum number of active calls admitted into the network at one ingress is constrained. Inside the network, no control mechanism is employed which grants any connection high degree of freedom to target any destination once admitted to the network.

To further explain the problem introduced when a non-tight *I-CAC* is employed, we present a simple example. A CAC unit is installed at node A of the network presented in Figure 5.4. This unit limits the maximum number of active connections entering the network at A to a preconfigured value, say 10. The estimated traffic demand in number of connections is given as follows.

$$\begin{array}{c} \\ \mathrm{A} \end{array} \begin{array}{c} \begin{array}{ccc} \mathrm{C} & \mathrm{D} & \mathrm{E} \end{array} \\ \left[\begin{array}{ccc} 6 & 4 & 0 \end{array}\right] \end{array} \tag{5.1}$$

This says that 6 connections go to destination C, the other 4 go to destination D, and no connection is directed to destination E. The given demand is only an estimation. In reality, it is not guaranteed that only 6 connections go to C, 4 go to D, and 0 go to E because there is no control inside the network to restrain the distribution of traffic flows. For example, it might happen that 3 go to C, 5 go to D, and 2 go to E. Other combinations different from the given values are also possible. This implies that if the network is dimensioned on the basis of the given demand only, that assumes 4 connections are going to D, all active connections directed to D might suffer

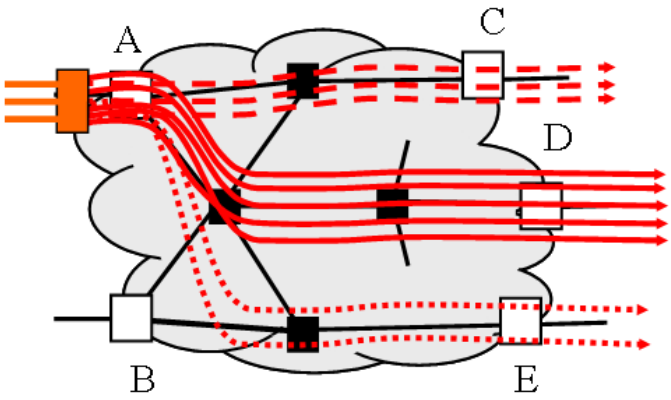


Figure 5.4: *I-CAC* with possible flow distribution.

from quality degradation if in reality 5 connections are destined to D. To assure satisfactory performance, enough resources should be provided within the network whenever needed. In this work, we propose an economical solution for such a problem that is most likely to occur in real networks. We study the statistical behavior of traffic flows inside an IP network and evaluate the deviation from the given demand values. To assure high robustness, we add extra capacity to the network links so they can accommodate additional flows with desired QoS.

For our analysis, we assume that each call requires a guaranteed effective bit rate R to assure a consistent quality level. There exist several formulae to compute R. Lindberger's formula [Lin94] [Lin99], for example, is a quite general and accurate approach for effective bit rate calculations. However, it does not consider the delay constraint, which is very critical in IP networks. In this thesis, we develop new capacity assignment methods that are specifically tailored for realtime services over IP networks where network delay is a significant factor in the methods (see Chapter 3 and Chapter 4). However, for the purpose of this chapter, we will keep it general as to which method is used.

5.3 Network Provisioning

5.3.1 Not Robust Approach

A direct approach for network provisioning is the one applied in circuit-switched networks. It basically depends on a fixed demand matrix and assumes only a fixed number of calls from a source to its targeted destination. This classical way works sufficiently well for PSTN or ATM networks. In IP networks, no complex resource control mechanisms are commonly deployed. This provides high freedom for the actual traffic to largely deviate from the given demand as demonstrated earlier and follow any distribution inside the network. Such traffic deviation causes dramatic degradation in quality. Therefore, this classical way of planning is not robust to traffic deviation and thus unsuitable for IP networks. We will still consider this classical approach for comparison purposes and refer to it hereinafter as *Not Robust*.

5.3.2 Strictly Robust Approach

A strictly robust network is guaranteed when network planners account for the ultimate degree of traffic deviation for the total traffic allowed into the network. We should then consider the extreme cases when the maximum traffic volume is directed to each destination.

Referring to the above example, this implies that we should dimension the link capacities based on the worst case scenarios when the total number of connections, which is 10, target C or D or E. As a result, the new traffic matrix becomes

$$\begin{array}{c} \\ \text{A} \end{array} \begin{array}{c} \begin{array}{ccc} \text{C} & \text{D} & \text{E} \end{array} \\ \left[\begin{array}{ccc} 10 & 10 & 10 \end{array} \right] \end{array} \tag{5.2}$$

This way, the planned link capacities can amply serve the active connections at all times irrespective of the way they are distributed in the network. Nevertheless, this approach imposes extremely high capacity requirements especially in the case of large-scale networks. We refer to this approach hereinafter as *Strictly Robust*.

At this point, we think of the need for a solution that offers a trade-off between network resources and performance. This can be possible if the given traffic demand is not fully trusted contrary to *Not Robust* and yet not fully distrusted contrary to *Strictly Robust*. In the following section, we present a novel approach that provides such a tradeoff solution.

5.3.3 Statistically Robust Approach

Our main intention is to provide a sufficiently robust network and perform efficient dimensioning. To fulfill this objective, we have first to determine the statistical distribution of the traffic volume directed to the different destinations. This is achieved by making proper use of the given traffic demand to statistically model the behavior of traffic flows inside the uncontrolled network. Referring to the above example, we need to evaluate the probability distribution function of the number of active connections directed to C, D, and E. With a desired probability value, we find out the maximum number of active connections that are destined to each of the destination nodes and use it in our planning process. We refer to this approach hereinafter as *Statistically Robust*.

The *Statistically Robust* approach comprises

1. obtaining a measure of the traffic that defines its statistical characteristics,
2. setting a quality measure that defines the desired degree of robustness,
3. allocating the link capacity share depending on results of step 1 and step 2.

Accomplishing step 3 may be done by determining a capacity margin that is added to the "non-robust" link capacity. In step 1, the measure that defines the traffic characteristics is the mean and variance of the traffic on the links. In step 2, the quality measure is selected and used with the measure from step 1 in step 3.

To obtain the statistical distribution of traffic inside the network, we start our analysis with the one-source-multiple-destination scenario. The expected traffic demand offered from source S to all destinations D_n, $n = 1, 2, ..., N$, is given as

$$\begin{array}{c} \\ \mathrm{S} \end{array} \begin{array}{c} \begin{array}{cccc} \mathrm{D}_1 & \mathrm{D}_2 & \ldots & \mathrm{D}_N \end{array} \\ \left[\begin{array}{cccc} A_1 & A_2 & \ldots & A_N \end{array} \right], \end{array} \tag{5.3}$$

where A_n represents the mean traffic load in Erlang offered from S to D_n in the busy hour. The total traffic volume A at source S is computed as

$$A = \sum_{n=1}^{N} A_n. \tag{5.4}$$

Depending on our network model presented in Section 5.2, a call admission control unit is placed at the ingress of the network. The task of this unit is to bound the maximum number of active connections allowed into the network. The maximum number of connections is denoted as $\widehat{K}$ and it can be computed by numerically inverting Erlang B formula which takes A and the desired call blocking probability as input variables.

In traditional telecommunication networks, a Poisson system is assumed where calls arrive according to a Poisson process and the service times have a negative exponential distribution. Assuming such a process, the resulting distribution of admitted traffic arrivals is a truncated Poisson process since the number of calls that enter the network at each ingress is limited. The nature of this distribution may still be preserved with a different mean value for traffic arrivals destined to destination D_n. For higher robustness, we consider the case when the maximum allowed number of calls is active and compute the distribution of the active calls (out of the $\widehat{K}$) that are directed to any destination D_n. For this case, the truncated Poisson process might not apply anymore. In the sequel, we attempt at evaluating the proper distribution of the active calls inside the network.

5.3.3.1 Traffic Distribution

We intuitively assume that calls are started independently of each other and that each call k has the following probability $P_{k,n}$ to target D_n.

$$P_{k,n} = P_n = \frac{A_n}{A} \quad 1 \leq k \leq \widehat{K}. \tag{5.5}$$

By this, we are able to make supplementary benefit from the given estimates of traffic demand to statistically model the system. For example, if one-source-two-destination scenario is given where A_1 = 6 Erlang and A_2 = 4 Erlang, we assume that a newly arriving call is destined to D_1 with a probability of 0.6 and to D_2 with a probability of 0.4. We note that empirical probability values of the possible destinations of a given call can be more easily obtained in enterprise networks where traffic statistics are collected. For example, an employee at a given branch connects to other branches of the enterprise or to external networks with certain probabilities.

In Figure 5.5, we show the statistical representation of the established connections between S and the N destinations. Note that the traffic demand from S to each D_n is treated independently.

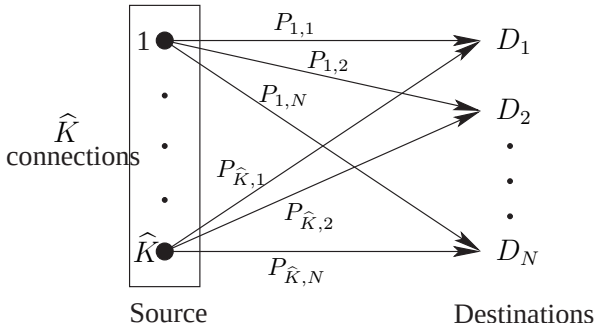


Figure 5.5: Statistical representation of $\widehat{K}$ calls between source S and its destinations D_n.

Let K_n denote the number of connections destined to D_n out of the $\widehat{K}$ connections. Then,

$$K_n = \sum_{k=1}^{\widehat{K}} I_{k,n}, \tag{5.6}$$

where $I_{k,n}$ is a Bernoulli random variable having the following probabilities.

$$P\{I_{k,n} = 1\} = P_n, \tag{5.7}$$
$$P\{I_{k,n} = 0\} = 1 - P_n. \tag{5.8}$$

$I_{k,n}$ serves as an indicator that tells whether the connection k is destined to D_n. Based on the central limit theorem, for $\widehat{K}$ large enough, K_n follows a normal distribution with mean μ_{K_n} and variance $\sigma^2_{K_n}$ computed as:

$$\mu_{K_n} = \widehat{K} \cdot P_n, \tag{5.9}$$
$$\sigma^2_{K_n} = \widehat{K} \cdot P_n \cdot (1 - P_n). \tag{5.10}$$

The normal distribution estimation of K_n gets more accurate as $\widehat{K}$ increases. Figure 5.6 shows that $\widehat{K} = 50$ leads already to accurate estimation. Note also that as $\widehat{K}$ increases, the mean and variance of K_n increase as well. Such observation agrees with (5.9) and (5.10).

Now that the distribution of the number of active connections directed to each D_n is known, we can dimension the network properly by accounting for traffic demand variability. To do so, we define a capacity margin M_{C} such that

$$P\{C_{n,\text{required}} > C_{n,\text{planned}} + M_{\mathrm{C}}\} \leq \varepsilon, \tag{5.11}$$
$$P\{K_n \cdot R > C_{n,\text{planned}} + M_{\mathrm{C}}\} \leq \varepsilon, \tag{5.12}$$

where $C_{n,\text{required}}$ is the capacity required to serve all active flows K_n on the links constituting the path from S to D_n (K_n is a random variable), $C_{n,\text{planned}}$ is the capacity calculated based on the given traffic demand, and ε is a parameter that determines the tolerance of quality degradation or

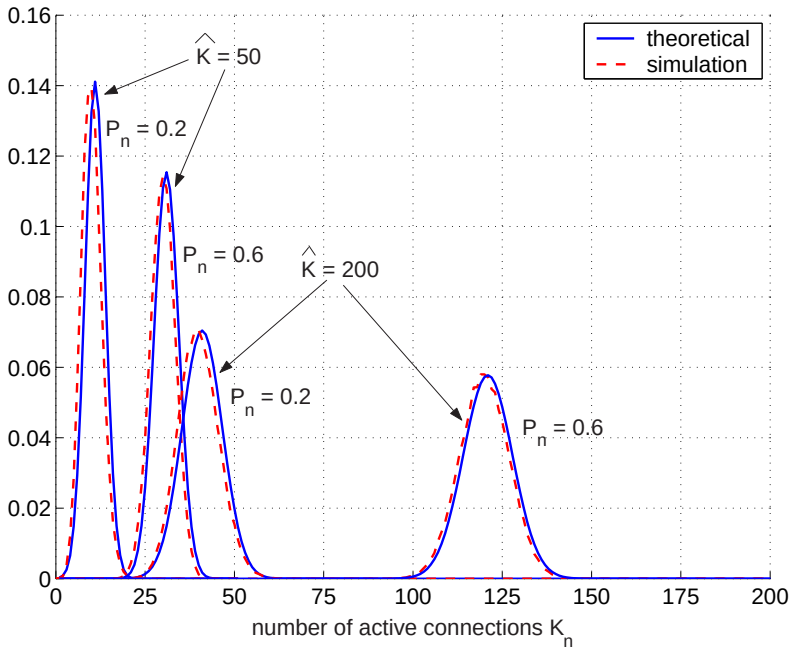


Figure 5.6: Probability distribution function of K_n for $\widehat{K} = 50$ and 200, and $P_n = 0.2$ and 0.6.

lack of robustness for the active connections in the network. Since K_n is shown to be normally distributed, then M_{C} can be numerically calculated using the following equation.

$$Q\left(\frac{M_{\mathrm{C}} + C_{n,\mathrm{planned}} - \widehat{K}\cdot P_n \cdot R}{R\cdot\sqrt{\widehat{K}\cdot P_n\cdot(1-P_n)}}\right) \;\leq\; \varepsilon, \tag{5.13}$$

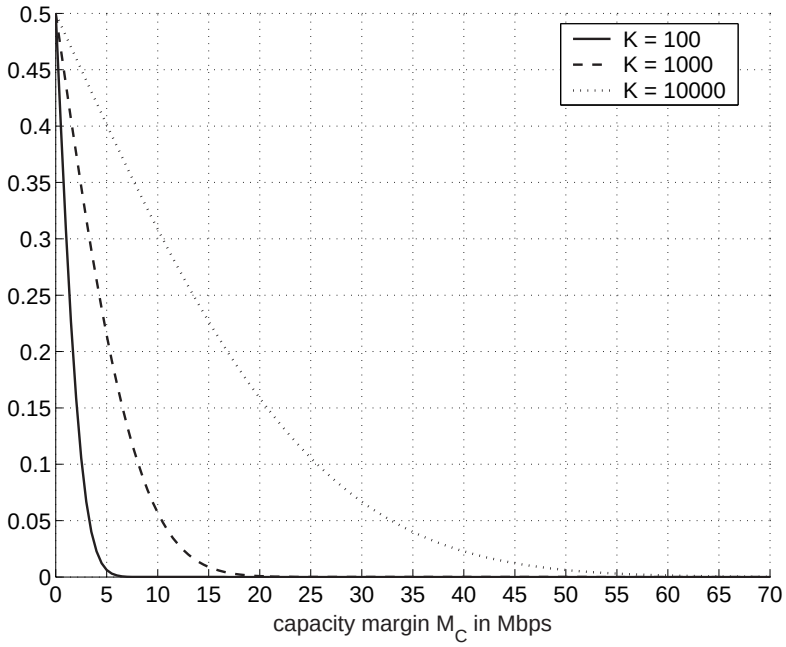
where $Q(x)$ is the Gaussian error integral known as the Q function and is given by

$$Q(x) \;=\; \frac{1}{2}\left(1 - \mathrm{erf}\left(\frac{x}{\sqrt{2}}\right)\right), \tag{5.14}$$

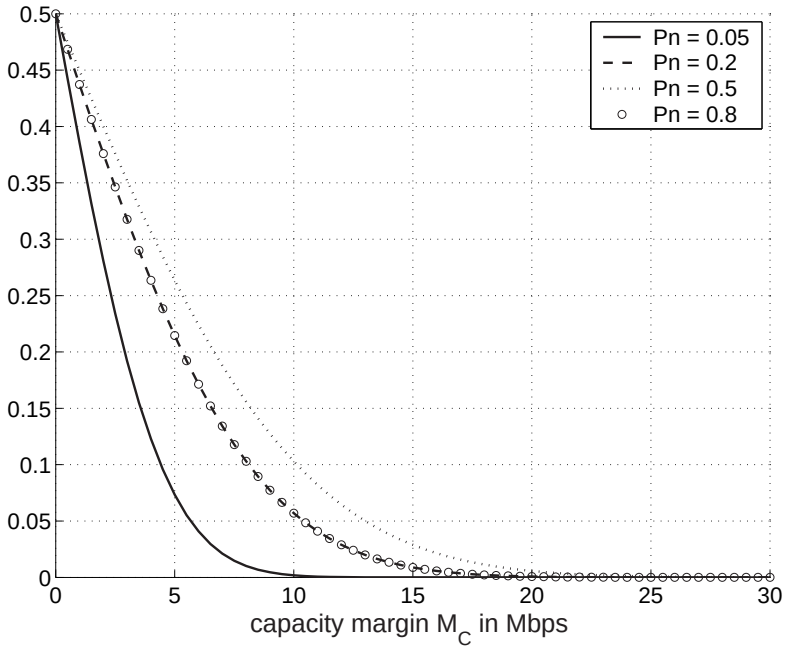
where $\mathrm{erf}(x)$ is the error function given by

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{(-u^2)}du. \tag{5.15}$$

Figure 5.7 plots ε in terms of M_{C} for selected values of $\widehat{K}$ and P_n, where $C_{n,\mathrm{planned}}$ is calculated according to the mean value of K_n and R is set to 0.5 Mbps. In Figure 5.7a, we observe that more capacity margin is required for the same tolerance degree as the maximum number of admitted connections into the network increases. If $\widehat{K} = 1000$ and $\varepsilon = 5\%$, the required capacity margin is $M_{\mathrm{C}} = 10$ Mbps. While if $\widehat{K} = 10000$ for the same ε, the required capacity margin is $M_{\mathrm{C}} = 33$ Mbps. We note, however, that $C_{n,\mathrm{planned}} = 100$ Mbps for $\widehat{K} = 1000$, and 1000 Mbps for $\widehat{K} = 10000$, i.e. $C_{n,\mathrm{planned}}$ increased 10 times for a 10-time increase in $\widehat{K}$, while the required M_{C} increased just 3.3 times.



(a)



(b)

Figure 5.7: Tolerance degree ε with respect to capacity margin M_{C}. (a) $P_n = 0.2$ and varying $\widehat{K}$. (b) $\widehat{K} = 1000$ and varying P_n.

In regards to Figure 5.7b, a similar trend applies for increasing P_n as long as P_n stays less than 0.5. If P_n exceeds 0.5, then higher values of P_n correspond to less capacity margin for the same tolerance degree. The curve corresponding to $P_n = 0.8$ falls below that of $P_n = 0.5$ and exactly coincides with the curve of $P_n = 0.2$. This observation is expected since $C_{n,\text{planned}} = \widehat{K} \cdot P_n \cdot R$, so (5.13) for $P_n = p$ and for $P_n = 1-p$, $0 \leq p \leq 1$, is identical. This behavior can be explained by the fact that $P_n = 0.5$ corresponds to the most uncertain scenario. If $P_n = 0$, we are certain that no calls are directed to destination D_n and thus we require no extra capacity margin to account for demand deviation. If $P_n = 1$, again we have a certain scenario that all calls are directed to D_n and thus no extra capacity margin is required since anyway we are accounting for the maximum number of calls. Therefore, for $P_n = 0.5$, the maximum capacity margin is required in order to account for the highest uncertainty in traffic demand.

In Figure 5.8, we plot the capacity margin in terms of P_n for various values of ε. We can observe that all curves of M_C reach their maximum at $P_n = 0.5$. This verifies our conclusion that the maximum margin is required for the maximum uncertainty and that is when $P_n = 0.5$. M_C has a value of 0 for the certain cases and that is when $P_n = 0$ or $P_n = 1$. Higher values of ε reduces the robustness demand on the network and so a lower value of M_C is needed. This is illustrated in the figure where curves corresponding to higher values of ε fall below those corresponding to lower values of ε.

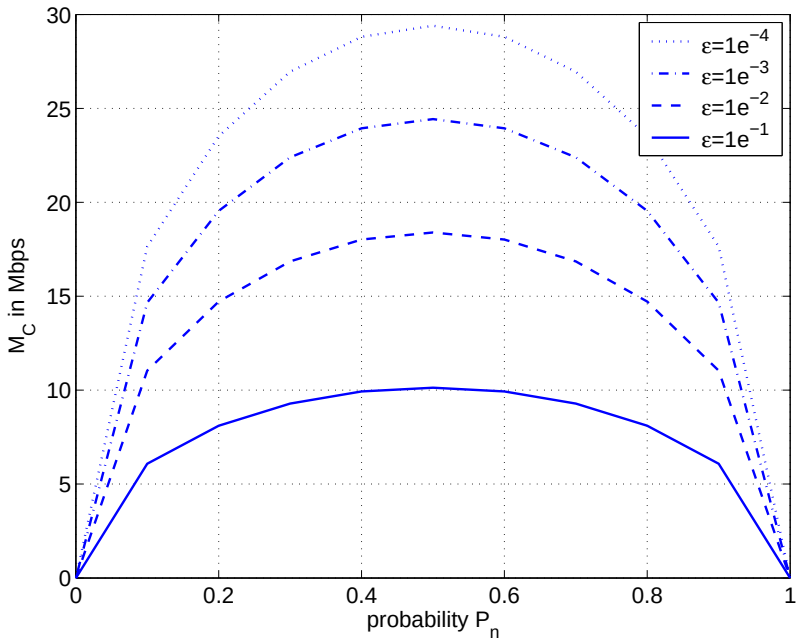


Figure 5.8: Capacity margin M_C versus P_n: $\widehat{K} = 1000$, and $R = 0.5$ Mbps.

5.4 Results and Analysis

The previously introduced approaches to network provisioning are now applied on a sample scenario network $N11$ whose topology is presented in Figure 5.9. Scenario $N11$ represents a DiffServ network with six boundary routers and five core routers. The boundary routers are marked dark in the figure. A CAC unit is assumed to be placed at their ingress to limit the

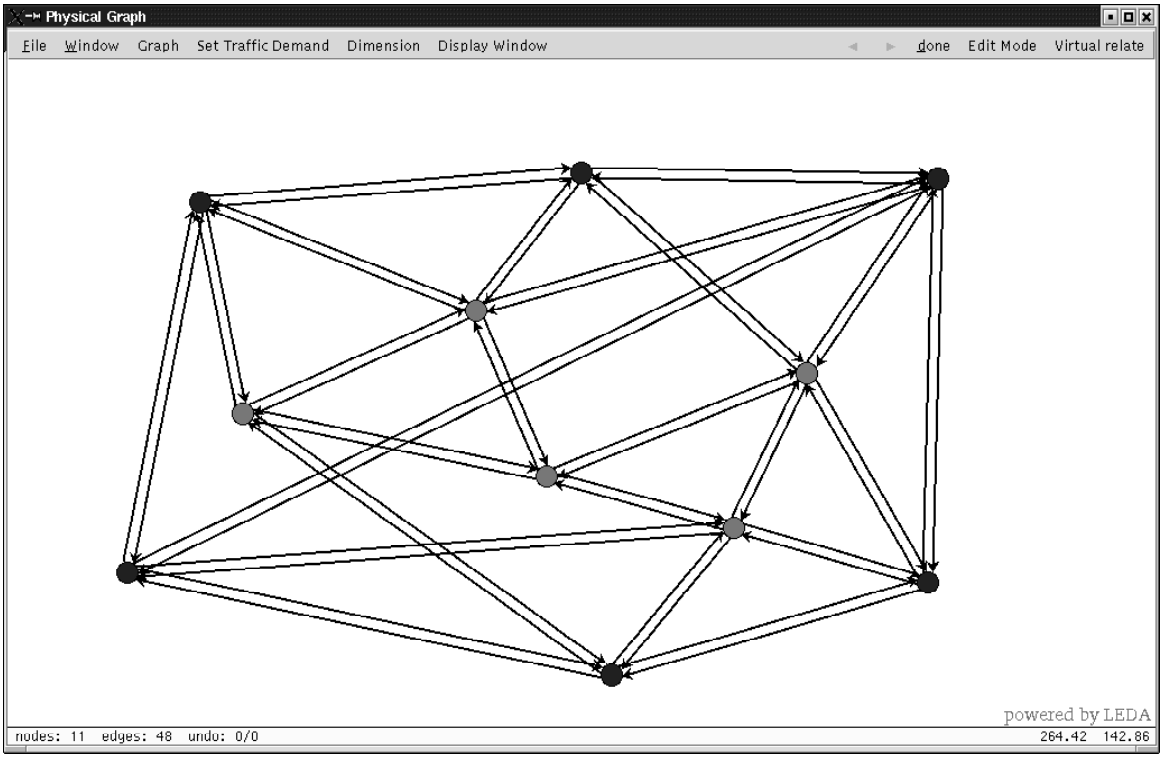


Figure 5.9: N11 network topology.

number of simultaneously active stream calls. At each boundary router, 500 users are connected and each user is assumed to generate 0.1 Erlang in the busy hour. The outgoing traffic of each boundary router is evenly distributed to the other boundary routers. The effective bit rate of each call is set to R and the target call blocking probability at each CAC unit is set to 1%.

According to Erlang B formula, we calculate the maximum number of active calls at each ingress. The resulting number of connections are then distributed according to the considered approach, whether *Not Robust*, *Strictly Robust*, or *Statistically Robust*. If the first approach is considered, active calls arriving at one ingress and directed to other boundary routers are distributed in proportion to the traffic ratio given in (5.5). Afterwards, connections are routed through the network to reach their destinations according to OSPF (open shortest path first) routing. At this point, the total number of active connections K_l routed on each link l is known and the link capacity C_l is calculated as

$$C_l = K_l \cdot R, \tag{5.16}$$

where R is the effective bit rate required for each call.

When *Strictly Robust* is applied to achieve a completely robust network, we dimension the network based on the assumption that all active calls at each ingress can be directed in total to each destination. We apply then OSPF routing to obtain the number of connections traversing each link and compute accordingly the corresponding link capacity as done in (5.16).

Finally, we apply the proposed *Statistically Robust* approach to assure high robustness against traffic demand deviation inside the network. Given ε, the capacity margin M_{C} for each link can

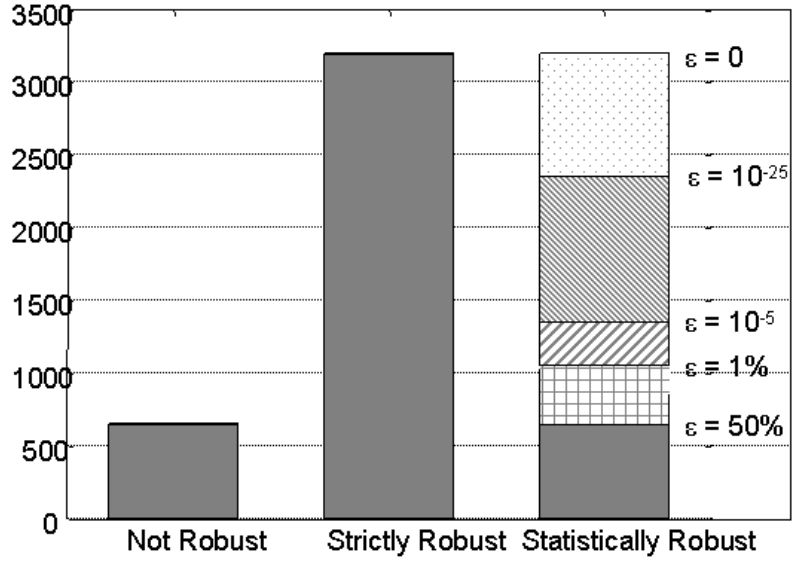


Figure 5.10: Total required link capacities C_{total} normalized with respect to R.

be calculated according to (5.13) where $C_{n,\text{planned}} = \mu_{K_n} \cdot R$. If single-path routing is used, traffic of one demand pair is routed through the same links. Then, we can compute one M_{C} for each demand pair as if one link is connecting the source and the destination. The obtained M_{C} is then added to all links constituting the routing path from source to destination.

However, since load sharing is performed in this example where calls are distributed evenly over equal-cost paths, then an alternative approach is followed. The maximum number of calls K_n associated to one demand pair is calculated according to (5.12) and based on the given ε. Routing is afterwards performed and the actual number of calls routed on each link is computed. Now, link capacities can be computed as a simple multiplication of the actual number of calls on one link by the effective bit rate (refer to (5.16)).

Figure 5.10 presents comparatively the dimensioning results of each approach. It is shown that *Not Robust* requires relatively the lowest amount of capacity but at the same time does not offer any kind of quality guarantees as soon as the actual traffic deviates from the given values. If a completely-robust network is desired, *Strictly Robust* is used and almost 5 times more capacity is required in comparison to the former case. However, for a slight smoothening of quality criteria, one is able to assure a highly robust network for relatively low costs (reduced capacity resources). *Statistically Robust* is shown as a reasonable tradeoff solution. If ε is set as low as 1%, *Statistically Robust* approach calls for around 1.5 times more capacity as compared to *Not Robust*. In total, the capacity margin required is $400 \cdot R$. For lower values of ε, the summation of all normalized link capacities is observed to increase slowly. The value of ε is reduced dramatically to 10^{-25} and the obtained value of the link capacity summation reaches $2350 \cdot R$. In this case, the total capacity margin is $1700 \cdot R$. Link capacity summation reaches the value of *Strictly Robust* when ε is set to 0.

5.5 Summary

For a successful migration from circuit-switched to packet-switched networks, the classical network planning process should be re-considered if still applicable to packet-switched networks. Circuit-switched networks offer high quality performance and availability to applications with stream traffic. To provide similar performance in packet-switched networks and IP networks in particular, proper network planning is a must. However, the available IP network architecture is not expected to meet the criteria unless complex resource control mechanisms are deployed. A variety of these mechanisms are realized in the literature but did not reach a status to be commonly implemented. Yet, they mostly require high costs. In this chapter, we handled the issue of network dimensioning for stream traffic under uncertain demands and absence of complex resource control mechanisms. We showed that a controlled overdimensioning is required to account for traffic demand deviation. In our analysis, we adopted the most practical and common resource control scenario where CAC is performed at the ingress of a DiffServ network. Based on this model, we statistically studied the distribution of traffic demand inside the uncontrolled network and introduced the concept of a capacity margin which is needed to account for a certain level of traffic demand variability in order to offer high degrees of robustness. We showed then by means of an example network scenario that provisioning a completely robust network requires unreasonable overdimensioning as compared to the classical way of dimensioning, which considers a fixed traffic matrix. However, if traffic demand distribution in the network is considered, we are able to offer a sufficiently reasonable tradeoff solution which prevails high level of robustness for relatively low capacity requirements.

6

Conclusions and Outlook

Realtime services are expected to play a crucial role in future IP networks, promising improved network efficiency, cost savings, and new revenue sources to network operators and service providers. For these services to succeed in future IP networks, stringent QoS criteria are required to preserve the same user experience of traditional voice and video networks. However, based on today's conventional IP technology, no QoS guarantees can be assured unless the network architecture is upgraded by deploying new mechanisms. These mechanisms should support predefined packet handling, bandwidth allocation, and call admission control. To carefully plan the network for these mechanisms to perform well, a solid understanding of the source traffic characteristics and its respective capacity requirements is necessary. Only then it is possible to keep up QoS by either reserving appropriate capacity shares or performing call admission control based on the allocated capacities.

In this work, we developed traffic models to accurately characterize traffic of realtime services and designed novel methods to determine efficient capacity requirements of these services. We then focused on network susceptibility to demand variability inside the network and proposed a robustness technique against this problem. In the following, we summarize the main points investigated in this dissertation.

Traffic Characterization and Capacity Requirements for Individual Sources

It is necessary that a traffic source either knows its characteristics and specifies the traffic profile to the network before sending, or a set of standard characteristics is assumed and pre-configured. Thus, accurate traffic models are needed, which allow to carefully describe the characteristics of the traffic and to derive its bandwidth requirements. In this work, we performed real-scenario measurements of various realtime applications with different coder settings. Based on the measurements results, we characterized the generated traffic using a common model to allow for fair comparison in our analysis and for fair treatment during network operation. We demonstrated that traffic flows may show different properties than the theoretical behavior especially when software-based applications are used. We then computed the needed capacities if a premium

and guaranteed quality is demanded for each traffic flow at any cost. This leads to the need for huge network resources that even increase dramatically when the generated traffic profile gets heavily distorted. Since it cannot be guaranteed that all sources generate smooth and predictable traffic, we illustrated the necessity for traffic policers to be installed for each traffic source. Traffic policers are given a certain profile to which they compare the arriving traffic. Traffic is allowed to pass through the policer unchanged only if it conforms to the given profile otherwise packets are discarded to force the incoming traffic to fit to the given profile. This limits the damage caused by bursty traffic to other active traffic that shares with it the same allocated resources (for example traffic belonging to the same class of service). By tuning traffic parameters of the adopted model, we investigated tradeoff possibilities between network costs in terms of capacity and resulting quality.

Capacity Assignment for Interactive Voice and Video Services

Due to their stringent QoS requirements, interactive voice and video services impose huge resource needs if no service impairment is tolerated. The allocated resources can only be efficiently used in extremely rare situations. If these situations are sacrificed, capacity requirements are drastically reduced. To this end, we focused on statistical evaluation of capacity needs of realtime services.

To handle this complex problem, we adopted the basic analysis-synthesis approach to zoom into the general network model (with multiple traffic classes) and examine the simple but nontrivial component. In our case, this component was the queue/buffer associated to the given service. We studied its operation in separation from other queues and interfering factors, and extended the study gradually from the buffer model to the node model, the path model, and finally the general network model using optimization techniques. In the extension from one model to the next, we considered the factors that exist in the next model and affect the operation of the current model. In regards to the interactive voice service, we designed our method at the buffer model based on the following concept: by providing quality guarantees for the most unlucky packets, we can implicitly provide guarantees to all other packets. We then extended this method gradually to include the effect of multiple traffic classes, multiple hop traversal, and multiple path existence. By means of simulations, we motivated this method and verified the analytical study. The benefits of this method are demonstrated using realistic network examples where we showed that the best tradeoff is obtained among other approaches: huge saving in capacity is obtained on the expense of slightly softening the QoS criteria.

As to the case of interactive video service, the previous method of interactive voice does not apply due to the variable behavior of video traffic. A new method was thus proposed. This method is based on an expression of the mean traffic rate, its standard deviation, and a parameter m. We motivated this method using simulations and computed the value of m for different number of aggregated videos. We demonstrated that m tends to converge to a common value when the number of different aggregated videos grows large. We then showed that this method is applicable to general forms of video traffic (whether MPEG or non-MPEG). We also studied the manner in which the requested delay threshold affects the capacity share needed. Finally, we demonstrated that the obtained results of m are almost independent of the quality level of the coded videos.

Network Planning Tool

For ease of planning, a software tool that performs all necessary operations in an automated manner is necessary especially in the case of large-scale networks. In this dissertation, we realized the network dimensioning problem in a generic network planning tool that scales to large networks with multiple services. We designed the tool with a three-layered architecture where each layer is assigned a number of tasks independently from other layers. To save execution time, we recommended some adaptation of the capacity assignment methods so time-consuming operations can be avoided. In the case of voice capacity assignment for example, we proposed a simple mapping that transforms end-to-end QoS parameters to per-hop QoS parameters in order to avoid the lengthy operation of optimization techniques. We tested the tool with the adapted version of the capacity assignment method on realistic network scenarios and showed highly satisfactory results. The accuracy of the obtained results was evaluated and shown to lead to slight overprovisioning.

Robust Network Dimensioning

IP network planning based on a given traffic demand becomes obsolete as soon as the actual traffic slightly deviates from the given values. It is highly probable that this be the case in IP networks with no deployed resource control mechanisms. Complex and comprehensive resource control mechanisms are still a matter of research. The most practical and common scenario is to have control units at the ingress of the IP network as pertained by the differentiated services architecture. In this dissertation, we showed that a controlled overdimensioning is required to account for traffic demand variability. This overdimensioning is represented by a capacity margin that is added to network link capacities so they can accommodate additional connections than planned. To compute the capacity margin, we studied the traffic demand distribution from one ingress to a given destination. This approach leads to a statistically robust network against traffic variability. In our computations, we made use of the given traffic demand matrix and intuitively assumed that the probability of an admitted connection to target one destination equates to the ratio of the mean traffic demand directed to this destination by the total traffic demand arriving at the ingress. Based on this assumption, it follows that the traffic demand directed from a given ingress to a given destination is normally distributed. By means of an example network scenario, we demonstrated that a completely robust network requires unreasonable overdimensioning while a statistically robust network leads to economical capacity values for satisfactory performance.

Outlook

In the course of this work, several interesting directions for further research have been identified, among which the following are of particular interest.

In the traffic characterization chapter, we found out that software-based clients might generate bursty traffic in loaded situations. For this problem, we proposed to deploy a token bucket policer for each client so as to assure that all admitted traffic fits to a given profile. In case token bucket policers are not deployed at the network edge, it is necessary that network planners consider the stochastic behavior of loaded clients while dimensioning the network where appropriate capacity shares for the given realtime service are allocated accordingly in order to avoid service degradation for active calls.

The capacity assignment method designed for interactive voice was applied on a realistic network model. Optimization techniques were used to find out the optimal combination of individual link capacities so as to achieve minimal capacity sums. During this process, fixed routing had been used. To achieve a more economical network solution, routing optimization would be jointly considered with capacity assignment. In regards to the capacity assignment method for interactive video, interesting research can still be done in terms of performing some analytical study on the method where the results and observations noted in our work are theoretically verified.

In the field of robust network dimensioning, the presented approach was applied to one call admission control scheme among several others. In [Rie03], a number of possible call admission control schemes are presented in an abstracted manner. For future research, we find it very interesting to apply our proposed approach to other CAC schemes and accordingly compute the capacity margin. The capacity margin is needed for achieving a statistically robust network. In addition, the proposed approach can be further extended. Traffic measurements can be done at different peak hours where statistics are collected and a more realistic empirical statistical model can be constructed. This model can then be used to derive more accurate capacity margins.

Finally, a software network planning tool was designed in this work so it accommodates new services. In this tool, we tested the proposed capacity assignment methods as well as the robust dimensioning approach which considers traffic demand variability. This tool can serve as a prototypical implementation for a commercial tool to be used for planning converged IP networks. This is of particular interest to service providers, network operators, and even enterprises who definitely started planning for their converged multiservice network. To commercialize this tool, faster processing and better graphical user interface should be provided. In addition, design methods should be implemented that assume a mixture of different voice and video coders in operation rather than one type of coders.

A

Abbreviations

ACELP	Algebraic Code-Excited Linear Prediction
ADPCM	Adaptive Differential Pulse Code Modulation
AF	Assured Forwarding
BA	Behavior Aggregates
B-frame	Bidirectionally-predictive picture
CA	Capacity Assignment
CAC	Call Admission Control
CAPEX	Capital Expenses
CBR	Constant Bit Rate
CB-WFQ	Class Based Weighted Fair Queuing
CCDF	Complementary Cumulative Distribution Function
CDF	Cumulative Distribution Function
CELP	Code Excited Linear Prediction
CIF	Common Intermediate Format
CL	Controlled Load
CNG	Comfort Noise Generator
CS-ACELP	Conjugate Structure Algebraic Code Excited Linear Prediction
DiffServ	Differentiated Services model
DS	Differentiated Services
DSCP	Differentiated Services Code Point
DSP	Digital Signal Processing
ECMP	Equal Cost Multi Path
EF	Expedited Forwarding
ETSI	European Telecommunications Standards Institute
F-ARIMA	Fractional Autoregressive Integrated Moving Average
FIFO	First In First Out
GOP	Group Of Picture
GPS	Global Positioning System
GS	Guaranteed Service

HDTV	High Definition Television
IETF	Internet Engineering Task Force
I-frame	Intra picture
IMPEX	Implementation Expenses
IntServ	Integrated Services model
IP	Internet Protocol
IS-IS	Intermediate System Intermediate System
ISP	Internet Service Provider
ITU	International Telecommunications Union
JPEG	Joint Photographic Experts Group
LD-CELP	Low Delay - Code Excited Linear Prediction
LEDA	Library for Efficient Data types and Algorithms
LRD	Long Range Dependence
MOS	Mean Opinion Score
MPEG	Moving Picture Experts Group
MPLS	Multi Protocol Label Switching
MP-MLQ	Multi Pulse Maximum Likelihood Quantization
MTU	Maximum Transfer Unit
NPL	Network Planning Library
OPEX	Operational Expenses
OSPF	Open Shortest Path First
PCM	Pulse Coded Modulation
PDF	Probability Density Function
P-frame	Predictive picture
PHB	Per Hop Behaviour
PMF	Probability Mass Function
PQ	Priority Queuing
QCIF	Quarter Common Intermediate Format
QoS	Quality of Service
RESV	Resource Reservation request
RIP	Routing Information Protocol
RSVP	Resource Reservation Protocol
RTP	Real Time Protocol
SLA	Service Level Agreement
SRD	Short Range Dependence
TB	Token Bucket
TC	Traffic Class
TCP	Transport Control Protocol
TOS	Type Of Service
VAD	Voice Activity Detector
VBR	Variable Bit Rate
VN	Virtual Node
VoIP	Voice over Internet Protocol
WDM	Wavelength Division Multiplexing
WFQ	Weighted Fair Queuing

Bibliography

[AGKT98] G. Apostolopoulous, R. Guerin, S. Kamat, and S. Tripathi. Quality of service based routing: A performance perspective. In *ACM SIGCOMM*, Vancouver, Canada, August 1998.

[AGL+98] D. Awduche, D. Gan, T. Li, G. Swallow, and V. Srinivasan. Extension to RSVP for traffic engineering. Technical report, Internet Draft. Internet Engineering Task Force (IETF), August 1998.

[ALS02] N. Ansari, H. Liu, and Y. Shi. On modeling MPEG video traffics. *On Modeling MPEG Video Traffics*, 48(4):337–347, 2002.

[Aqu00] Aquila home page, 2000. http://www.ist-aquila.org.

[BBC98] S. Blake, D. Black, and M. Carlson. An architecture for differentiated services. Request for comments RFC 2475. Technical report, Internet Engineering Task Force (IETF), December 1998.

[BCS94] R. Braden, D. Clark, and S. Shenker. Integrated services in the internet architecture: an overview. Request for comments RFC 1633. Technical report, Internet Engineering Task Force (IETF), June 1994.

[Ber99] P. Bertsekas. *Nonlinear Programming*. Athena Scientific, U.S.A, 2nd edition, 1999.

[BFY00] Y. Bernet, P. Ford, and R. Yavatkar. A framework for integrated services operation over diffserv networks. Request for comments RFC 2998. Technical report, Internet Engineering Task Force (IETF), November 2000.

[BGT93] C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shanon limit error-correcting coding and decoding: Turbo codes. In *IEEE International Conference on Communications (ICC)*, Geneva, May 1993.

[BO00] M. Baldi and Y. Ofek. End-to-end delay analysis of videoconferencing over packet switched networks. *IEEE/ACM Transactions on Networking*, 8(4):479–492, August 2000.

[BPR01] T. Bonald, A. Proutiere, and J. Roberts. Statistical performance guarantees for streaming flows using expedited forwarding. In *INFOCOM*, Anchorage, Alaska, April 2001.

[BVJP02] M. Büchli, D. De Vleeschauwer, J. Janssen, and G. Petit. Policing aggregates of voice traffic with the token bucket algorithm. In *IEEE International Conference on Communications (ICC)*, New York, USA, April/May 2002.

[BZB+97] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource reservation protocol (RSVP) - version 1 functional specification. Request for comments RFC 2205. Technical report, Internet Engineering Task Force (IETF), September 1997.

[CBC99] E. Cristian, E. Brococi, and A. Constantin. Multimedia oriented transport architectures and QoS management. Technical report, Institut Nationale des Telecommunications, No. 06-LOR, 1999.

[CDWB00] J. Cao, D. Davis, S. Wiel, and B.Yu. Time-varying network tomography. *J. of the American Statistical Association*, 2000.

[Cha94] C. Chang. Stability, queue length and delay of deterministic and stochastic queuing networks. *IEEE Transactions on Automatic Control*, 39(5):913–931, May 1994.

[CK02] C. Chuah and R. Katz. Characterizing packet audio streams from internet multimedia applications. In *IEEE International Conference on Communications (ICC)*, New York, USA, April/May 2002.

[CNRS98] E. Crawley, R. Nair, B. Rajagopalan, and H. Sandick. A framework for QoS-routing in the internet. Request for comments RFC 2386. Technical report, Internet Engineering Task Force (IETF), August 1998.

[Coh79] J. Cohen. The multiple phase service network with generalized processor sharing. *Acta Informatica*, 12:245–284, August 1979.

[Con02] A. Conway. A passive method for monitoring voice-over-IP call quality with ITU-T objective speech quality measurement methods. In *IEEE International Conference on Communications (ICC)*, New York, USA, April/May 2002.

[CR01] R. Cole and J. Rosenbluth. Voice over IP performance monitoring. *ACM SIGCOMM Computer Communication Review*, 31(2):9–24, April 2001.

[Cru91] R. Cruz. A calculus for network delay, Part I: Network elements in isolation. *IEEE Transactions on Information Theory*, 37(1):114–131, January 1991.

[Cru95] R. Cruz. Quality of service guarantees in virtual circuit switched networks. *IEEE Journal on Selected Areas in Communication*, 13(6):1048–1056, August 1995.

[CWYZ00] J. Cao, S. Wiel, B. Yu, and Z. Zhu. A scalable method for estimating network traffic matrices from link counts. Technical report, Bell Labs, 2000.

[Eck79] A. Eckberg. The single server queue with periodic arrival process and deterministic service times. *IEEE Transactions on Communications*, 27(3):556–562, March 1979.

[EG04] J. Eberspächer and J. Glasmann. *QoS-Architekturen und Ressourcenmanagement im Intranet*. Springer-Verlag, Germany, 2004.

[Eth01] Ethereal, a network protocol analyzer for unix and windows, 2001. http://www.ethereal.com.

[FGL+01] A. Feldmann, A. Greeberg, C. Lung, N. Reinolds, J. Rexford, and F. True. Deriving traffic demands for operational IP networks: Methodology and experience. *IEEE/ACM Transaction on Networking*, pages 265–279, June 2001.

[For96] ATM Forum. Traffic management specification version 4.0. Technical report, ATM Forum, February 1996.

[FR00] F. Fitzek and M. Reisslein. MPEG-4 and H.263 video traces for network performance evaluation, 2000. http://www-tkn.ee.tu-berlin.de/research/trace/trace.html.

[FR01] E. Fulp and D. Reeves. Optimal provisioning and pricing of differentiated services using QoS class promotion. In *Advanced Internet Charging and QoS Technology (ICQT)*, Vienna, Austria, September 2001.

[FWD+02] F. Le Faucher, L. Wu, B. Davie, S. Davari, P. Vaananen, R. Krishnan, P. Cheval, and J. Heinanen. Multi-protocol label switching (MPLS) support of differentiated services. Request for comments RFC 3270. Technical report, Internet Engineering Task Force (IETF), May 2002.

[GBH97] R. Guerin, S. Blake, and S. Herzog. Aggregating RSVP-based QoS requests. internet draft. Technical report, Internet Draft. Internet Engineering Task Force (IETF), November 1997.

[GGPR96] L. Georgiadis, R. Guérin, V. Peris, and R. Rajan. Efficient support of delay and rate guarantees in an internet. In *ACM SIGCOMM*, Palo Alto, CA, August 1996.

[Gla03] J. Glasmann. *Resourcemanagement für Echtzeitverkehre in Intranets*. Dissertation. Institute for Communication Networks, Technische Universität München, Munich, Germany, 2003.

[Gol00] O. Goldschmidt. ISP backbone traffic inference methods to support traffic engineering. In *Internet Statistics and Metrics Analysis Workshop*, San Diego, CA, December 2000.

[GW94] M. Garrett and W. Willinger. Analysis, modeling, and generation of self-similar VBR video traffic. In *ACM SIGCOMM*, London, UK, September 1994.

[Haj94] B. Hajek. A queue with periodic arrivals and constant service rate. In F. P. Kelly, editor, *Probability Statistics and Optimization – A Tribute to Peter Whittle*, pages 147–158. John Wiley and Sons, 1994.

[Haß01] G. Haßlinger. Quality-of-Service analysis for statistical multiplexing with Gaussian distributed and Autoregressive input. *Telecommunication Systems*, 16(3):315–334, 2001.

[HBH93] P. Humblet, A. Bhargava, and M. Hluchyj. Ballot theorems applied to the transient analysis of $nD/D/1$ queues. *IEEE/ACM Transactions on Networking*, 1(1):81–95, February 1993.

[HBW99] J. Heinanen, F. Baker, and W. Weiss. Assured forwarding PHB group. Request for comments RFC 2597. Technical report, Internet Engineering Task Force (IETF), June 1999.

[HDLK96] C. Huang, M. Devetsikiotis, I. Lambadaris, and A. Kaye. Modeling and simulation of self-similar variable bit rate compressed video: a unified approach. In *ACM SIGCOMM*, Palo Alto, CA, August 1996.

[HF02] G. Haßlinger and M. Fiedler. Why buffers in switching do not essentially improve QoS: an analytical case study for on/off source traffic. *Internet Performance and Control of Network Systems, SPIE*, 4865, 2002.

[HT03] G. Haßlinger and P. Takes. Real time video traffic characteristics and dimensioning regarding QoS demands. In *18th International Teletraffic Congress (ITC'18)*, Berlin, Germany, August/September 2003.

[Hui88] J. Hui. Resource allocation for broadband networks. *IEEE Journal on Selected Areas in Communication*, 6(9):1598–1608, 1988.

[Ins04] Insight. *The Telecommunications Industrial Review: An Anthology of Market Facts and Forecasts*. Insight Research Corporation, New Jersey, USA, 2004.

[IR99] M. Izquierdo and D. Reeves. A survey of statistical source models for variable-bit-rate compressed video. *Multimedia Systems*, 7, 1999.

[ISO91] Coding of moving pictures and associated audio for digital storage media up to about 1.5 Mbps. Technical report, ISO/IEC 11172, 1991.

[ISO94] Generic coding of moving pictures and associated audio information. Technical report, ISO/IEC 13818, 1994.

[ISO99] Coding of moving pictures and audio. Technical report, ISO/IEC 14496, 1999.

[ISO01] Multimedia content description interface. Technical report, ISO/IEC 15938, 2001.

[ITU72] Pulse code modulation (PCM) of voice frequencies. Technical report, ITU-T Recommendation G.711, November 1972.

[ITU90] ADPCM: 40, 32, 24, 16 kbps adaptive differential pulse code modulation. Technical report, ITU-T Recommendation G.726, December 1990.

[ITU92] Coding of speech at 16 kbps using low-delay code excited linear prediction. Technical report, ITU-T Recommendation G.728, September 1992.

[ITU93] Video codec for audiovisual services at p $\times$ 64 kbits. Technical report, ITU-T Recommendation H.261, March 1993.

[ITU96a] Annex A: C reference code, test signals and test sequences for the fixed point 5.3 and 6.3 kbps dual rate speech coder and for the silence compression scheme. Technical report, ITU-T Recommendation G.723.1, November 1996.

[ITU96b] Annex A: C source code and test vectors for implementation verification of the G.729 reduced complexity 8 kbps CS-ACELP speech coder. Technical report, ITU-T Recommendation G.729, November 1996.

[ITU96c] Coding of speech at 8 kbps using conjugate-structure algebraic-code-excited linear-prediction. Technical report, ITU-T Recommendation G.729, March 1996.

[ITU96d] Dual rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbps. Technical report, ITU-T Recommendation G.723.1, March 1996.

[ITU96e] Traffic control and congestion control in B-ISDN. Technical report, ITU-T Recommendation I.371, May 1996.

[ITU96f] Video coding for low bit rate communication. Technical report, ITU-T Recommendation H.263, March 1996.

[ITU97] Annex B: C source code and test vectors for implementation verification of the algorithm of the G.729 silence compression scheme. Technical report, ITU-T Recommendation G.729, August 1997.

[ITU98a] Annex D: 6.4 kbps CS-ACELP speech coding algorithm. Technical report, ITU-T Recommendation G.729, September 1998.

[ITU98b] Annex E: 11.8 kbps CS-ACELP speech coding algorithm. Technical report, ITU-T Recommendation G.729, September 1998.

[ITU98c] Video coding for low bit rate communication. Technical report, ITU-T Recommendation H.263, January 1998.

[ITU99a] Annex H: Variable bit rate LD-CELP operation mainly for DCME at rates less than 16 kbps. Technical report, ITU-T Recommendation G.728, May 1999.

[ITU99b] Application of the E-model: A planning guide. Technical report, ITU-T Recommendation G.108, September 1999.

[ITU00] One-way transmission time. Technical report, ITU-T Recommendation G.114, May 2000.

[JNP99] V. Jacobson, K. Nichols, and K. Poduri. An expedited forwarding. Request for comments RFC 2598. Technical report, Internet Engineering Task Force (IETF), June 1999.

[JNP00] V. Jacobson, K. Nichols, and K. Poduri. The virtual wire ‘per-domain behavior’: Analysis and extensions. Technical report, Internet Engineering Task Force (IETF), July 2000.

[JS99] W. Jiang and H. Schulzrinne. QoS measurement of internet real-time multimedia services. Technical report, Columbia University, December 1999.

[KKS01] V. Kumar, M. Korpi, and S. Sengodan. *IP Telephony with H.323: Architectures for Unified Networks and Integrated Services*. John Wiley and Sons, New York, 2001.

[Kle75] L. Kleinrock. *Queueing Systems: Computer Applications*, volume 2. John Wiley and Sons, New York, 1975.

[KM98] M. Krunz and A. Makowski. Modeling video traffic using M/G/∞ input processes: A compromise between markovian and LRD models. *IEEE J. on Selected Areas in Communications*, 16(5):733–748, June 1998.

[Ksn01] Ksnuffle - a network packet sniffer for kde, 2001. http://www.quaking.demon.co.uk/ksnuffle.

[KT00] M. Karam and F. Tobagi. On traffic types and service classes in internet. In *GLOBECOM*, San Francisco, CA, USA, December 2000.

[KT01] M. Karam and F. Tobagi. Analysis of the delay and jitter of voice traffic over the Internet. In *INFOCOM*, Anchorage, Alaska, USA, April 2001.

[LED02] Library for efficient data types and algorithms (LEDA), 2002. Algorithmic Solutions Software Gmbh. http://www.algorithmic-solutions.com/enleda.htm.

[LH92] J. Labourdette and G. Hart. Blocking probabilities in mutlitraffic loss systems: Insensitivity, asymptotic behaviour and approximations. *IEEE Transactions on Communications*, 40(8):1355–1366, August 1992.

[Lin94] K. Lindberger. Dimensioning and design methods for integrated ATM networks. In *14th International Teletraffic Congress (ITC'14)*, pages 897–906, Antibes Juan-les-Pins, France, June 1994.

[Lin99] K. Lindberger. Balancing quality of service, pricing and utilisation in mutliservice networks with stream and elastic traffic. In *16th International Teletraffic Congress (ITC'16)*, Edinburgh, UK, June 1999.

[LR98] T. Li and Y. Rekhter. Provider architecture for differentiated services and traffic engineering (PASTE). Request for comments RFC 2430. Technical report, Internet Engineering Task Force (IETF), October 1998.

[LVR+03] C. López, M. Veiga, R. Rodríguez, A. Suárez, and D. Teijeiro. Effect of the generation of MPEG-frames within a GOP on queuing performance. In *18th International Symposium on Computer and Information Sciences*, Antalya, Turkey, November 2003.

[Mar03] V. Marbukh. A scenario based framework for robust network provisioning. In *18th International Teletraffic Congress (ITC'18)*, Berlin, Germany, August 2003.

[MAS+88] B. Maglaris, D. Anastassiou, P. Sen, G. Karlson, and J. Robbins. Performance models for statistical multiplexing in packet video communications. *IEEE Transactions on Communications*, 36:834–843, 1988.

[MCPA01] S. Mohamed, F. Cervantes-Perez, and H. Afifi. Integrating network measurements and speech quality subjective scores for control purposes. In *INFOCOM*, Anchorage, Alaska, USA, April 2001.

[MKM04] M. Menth, S. Kopf, and J. Milbrandt. A perfomrance evaluation framework for network admission control methods. In *IEEE Network Operations and Management Symposium (NOMS)*, Seoul, South Korea, April 2004.

[MMK04] M. Menth, J. Milbrandt, and S. Kopf. Impact of routing and traffic distribution on the performance of network admission control. In *9th IEEE Symposium on Computers and Communications (ISCC)*, Alexandria, Egypt, June/July 2004.

[MT02] J. Matta and A. Takeshita. End-to-end voice over IP quality of service estimation through router queuing delay monitoring. In *GLOBECOM*, Taipei, Taiwan, November 2002.

[MTK03] A. Markopoulou, F. Tobagi, and J. Karam. Assessing the quality of voice communications over internet backbones. *IEEE Transactions on Networking*, 11(5):747–760, October 2003.

[MTS+02] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot. Traffic matrix estimation: Existing techniques and new directions. In *ACM SIGCOMM*, Pittsburgh, PA, August 2002.

[MWKB99] M. Mandjes, K. van der Wal, R. Kooij, and H. Bastiaansen. End-to-end delay models for interactive services on a large-scale IP network. In *IFIP Workshop on Modeling and Evaluation of ATM/IP Networks*, Antwerp, Belgium, June 1999.

[NPGI99] A. Nilson, M. Perry, A. Gersht, and V. Iversen. On multi-rate Erlang-B computations. In *16th International Teletraffic Congress (ITC'16)*, Edinburgh, UK, June 1999.

[NPL02] Network planning library (NPL), 2002.

[NQBM99] R. Nunez-Queija, H. van den Berg, and M. Mandjes. Performance evaluation of strategies for integration of elastic and stream flows. In *16th International Teletraffic Congress (ITC'16)*, Edinburgh, UK, June 1999.

[OS91] T. Ott and J. Shantikumar. On a buffer problem for packetized voice with n-periodic strongly interchangeable input processes. *Journal on Applied Probability*, pages 630–646, 1991.

[PG93] A. Parekh and R. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The single-node case. *IEEE/ACM Transactions on Networking*, 1(3), June 1993.

[PG94] A. Parekh and R. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The multiple-node case. *IEEE/ACM Transactions on Networking*, 2(2), April 1994.

[PV03a] G. Prasanna and A. Vishwanath. A reformulation of network design: Replacing the uncertain traffic matrix by a web of linear relations, and its implications. In *Optical Fiber Communication Conference (OFC)*, Atlanta, Georgia, USA, March 2003.

[PV03b] G. Prasanna and A. Vishwanath. Traffic constraints instead of traffic matrices: Capabilities of a new approach to traffic characterization. In *18th International Teletraffic Congress (ITC'18)*, Berlin, Germany, August 2003.

[RASHB03] L. Roychoudhuri, E. Al-Shaer, H. Hamed, and G. Brewster. Audio transmission over the internet: Experiments and observations. In *IEEE International Conference on Communications (ICC)*, Anchorage, Alaska, USA, May 2003.

[RBF02] A. Riedl, T. Bauschert, and J. Frings. A framework for multi-service IP network planning. In *Networks*, 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 *18th International Teletraffic Congress (ITC'18)*, Berlin, Germany, August 2003.

[Rie03] A. Riedl. *Routing Optimization and Capacity Assignment in Multi-Service IP Networks*. Dissertation. Institute for Communication Networks, Technische Universität München, Munich, Germany, 2003.

[RMV96] J. Roberts, U. Mocci, and J. Virtamo. *Broadband Network Teletraffic: Final Report of Action COST 242*. Springer-Verlag, Berlin, Germany, 1996.

[RV91] J. Roberts and J. Virtamo. The superposition of periodic cell arrival processes in an ATM multiplexer. *IEEE Transactions on Communications*, 39:298–303, February 1991.

[RVC01] E. Rosen, A. Viswanathan, and R. Callon. Multi-protocol label switching architecture. Request for comments RFC 3031. Technical report, Internet Engineering Task Force (IETF), January 2001.

[SD04] S. Sharafeddine and Z. Dawy. A capacity margin for IP networks with QoS constraints and uncertain demands. In *9th IEEE Symposium on Computers and Communications (ISCC)*, Alexandria, Egypt, June/July 2004.

[SD05] S. Sharafeddine and Z. Dawy. Capacity assignment for video trac in multiservice IP networks with statistical QoS guarantees. In *10th IEEE Symposium on Computers and Communications (ISCC)*, Cartagena, Spain, June/July 2005.

[Sha04] S. Sharafeddine. Implementation of the almost guaranteed dimensioning strategy in integrated IP networks. In *9th IEEE Symposium on Computers and Communications (ISCC)*, Alexandria, Egypt, June/July 2004.

[SKD04] S. Sharafeddine, N. Kongtong, and Z. Dawy. Capacity allocation for voice over IP networks using maximum waiting time models. In *11th Intenational Conference on Telecommunications (ICT). Lecture Notes on Computer Science*, Fortaleza, Brazil, August 2004.

[SPG97] S. Shenker, C. Partridge, and R. Guerin. Specification of guaranteed quality of service. Request for comments RFC 2212. Technical report, Internet Engineering Task Force (IETF), September 1997.

[SRB05] S. Sharafeddine, A. Riedl, and T. Bauschert. Network capacity optimization for latency sensitive traffic in multi-service IP networks. In *19th International Teletraffic Congress (ITC'19)*, Beijing, China, August/September 2005.

[SRGT03] S. Sharafeddine, A. Riedl, J. Glasmann, and J. Totzke. On traffic characteristics and bandwidth requirements of voice over IP applications. In *8th IEEE Symposium on Computers and Communications (ISCC)*, Turkey, June/July 2003.

[SRT03] S. Sharafeddine, A. Riedl, and J. Totzke. A dimensioning strategy for almost guaranteed quality of service in voice over IP networks. In *6th IEEE International Conference on High Speed Networks and Multimedia Communications (HSNMC)*, Estoril, Portugal, July 2003.

[SW86] K. Sriram and W. Whitt. Characterizing superposition arrival processes in packet multiplexers for voice and data. *IEEE Journal on Selected Areas in Communications*, 4(6):833–846, September 1986.

[Tcp01] Tcpdump, 2001.

[TW98] C. Tebaldi and M. West. Bayesian inference of network traffic using link count data. *J. of the American Statistical Association*, pages 557–573, June 1998.

[Unt04] L. Untereiner. *Video Traffic Modeling and Dimensioning in IP Networks*. Diploma Thesis. Institute for Communication Networks, Technische Universität München, Munich, Germany, 2004.

[Var96] Y. Vardi. Network tomography: Estimating source-destination traffic intensities from link data. *J. of the American Statistical Association*, 91(433):365–377, March 1996.

[VG02] S. Vaton and A. Gravey. Iterative Bayesian estimation of network traffic matrices in the case of bursty flows. In *ACM SIGCOMM Workshop on Internet Measurement*, Marseille, France, November 2002.

[Wro97] J. Wroclawski. Specification of the controlled-load-network element service. Request for comments RFC 2211. Technical report, Internet Engineering Task Force (IETF), September 1997.

[XN99] X. Xiao and L. Ni. Internet QoS: A big picture. *IEEE Network*, 13(2):8–18, March 1999.

[ZRLD03] Y. Zhang, M. Roughan, C. Lund, and D. Donoho. An information-theoretic approach to traffic matrix estimation. In *ACM SIGCOMM*, Karlsruhe, Germany, August 2003.