Efficient Broadband

Performance Analysis and Design of ATM Networks

Borut Stavrov

Delft University Press, 1999
Efficient Broadband

Performance Analysis and Design of ATM Networks

PROEFSCHRIFT

ter verkrijging van de graad doctor
aan de Technische Universiteit Delft
op gezag van de Rector Magificus prof.ir. K.F. Wakker,
in het openbaar te verdedigen ten overstaan van een commissie,
 door het College voor Promoties aangewezen,

op maandag 11 oktober 1999 te 16.00 uur

doors

BORUT STAVROV

elektrotechnisch ingenieur,
geboren te Skopje
Dit proefschrift is goedgekeurd door de promotor:
Prof.dr. F.C. Schoute

Samenstelling Promotiecommissie:

Rector Magnificus, voorzitter,
Prof.dr. F.C. Schoute, Technische Universiteit Delft, promotor
Prof.dr.ir. J. Biemond, Technische Universiteit Delft
Prof.dr.ir. P.F.A. van Mieghem, Technische Universiteit Delft
Prof.dr.ir. O.J. Boxma, Technische Universiteit Eindhoven
Prof.dr. J. van der Wal, Universiteit van Amsterdam
Prof.ir. J.W.J. van Till, Technische Universiteit Delft
Prof.ir. F. van den Dool, Technische Universiteit Eindhoven
For my late grandmother Adriana who grew me up.
Who showed me the beauty of the jewel of knowledge
and taught me how to explore it.

Who loved me enough to let me be.
Contents

Summary ................................................................. 1

1 Introduction .......................................................... 3
   1.1 A glance at the future ......................................... 3
   1.2 Evolution of the communication networks ................. 6
   1.3 Asynchronous Transfer Mode ................................ 10
      1.3.1 ATM Protocol reference model ....................... 13
   1.4 Open issues in ATM networks .............................. 16
      1.4.1 Network design ....................................... 17
      1.4.2 Traffic Control ..................................... 18
   1.5 ATM performance modeling ................................. 19
   1.6 Scope and Outline of the thesis .......................... 20

2 Characterization of broadband traffic ......................... 23
   2.1 Broadband traffic classification .......................... 23
   2.2 The nature of broadband traffic .......................... 26
      2.2.1 Definitions and properties of stochastic self-similar processes 28
   2.3 Traffic modeling ........................................... 32
      2.3.1 Connection level models ........................... 32
      2.3.2 Burst level models .................................. 33
      2.3.3 Verification of the Burst level models ............. 42
      2.3.4 Cell level models .................................. 44
   2.4 Conclusion .................................................. 44

3 ATM resource sharing ........................................... 49
   3.1 CELL level analysis ....................................... 51
      3.1.1 ATM switching element architecture ................. 51
      3.1.2 The single outlet LDOLL queue ....................... 53
      3.1.3 Generalization of the LDOLL queue to multiple outlets 60
   3.2 Burst level analysis ....................................... 76
      3.2.1 Performance evaluation methods .......................... 76
      3.2.2 Burst level analysis of the LDOLL queue ............. 80
      3.2.3 The impact of Self-Similarity on the buffer performance 84
   3.3 Conclusion .................................................. 86
4 ATM network design
   4.1 Introduction .................................................. 90
   4.2 Problem formulation ........................................... 93
      4.2.1 Network model ............................................. 93
      4.2.2 Optimization problem formulation ..................... 95
   4.3 The solution approach .......................................... 97
      4.3.1 The design framework .................................... 98
      4.3.2 Topology optimization and Convergence control .......... 100
      4.3.3 VP mapping ................................................ 102
      4.3.4 Routing and Dimensioning .................................. 104
   4.4 BANDIT: a flexible ATM network design tool ................ 111
      4.4.1 The Graphical User Interface ............................ 112
   4.5 Preliminary experimental results ............................... 117
      4.5.1 Network definition ........................................ 118
      4.5.2 Experimental results ..................................... 120
   4.6 Conclusion ...................................................... 124

5 Connectionless enterprise ATM overlay network design .......... 127
   5.1 Network architecture and architecture issues ............... 128
   5.2 Network modeling and problem formulation .................... 131
   5.3 Solution procedure ............................................ 134
      5.3.1 Algorithm implementation ................................ 136
      5.3.2 Routing and Link dimensioning ........................... 137
   5.4 Experimental results ........................................... 138
      5.4.1 Quality of the optimization procedure .................. 139
      5.4.2 CLS costs vs. link costs ................................ 141
   5.5 Conclusion ...................................................... 144

6 Conclusions ........................................................ 147
   6.1 Claims .......................................................... 148
   6.2 Recommendations for future research ........................ 150

Bibliography .......................................................... 153

List of Symbols ...................................................... 163

List of Abbreviations ................................................ 167

Samenvatting .......................................................... 171

Acknowledgements .................................................... 173

Curriculum Vitae ..................................................... 175
Summary

In the last few years the interest in the Global Information Infrastructure (GII) has increased enormously, both from the industry and the general public, primarily due to the ever-growing popularity of the Internet and the rapid development of novel distributed-network technologies, like Java. The central subject of this thesis is the efficient design of the future Broadband Integrated Services Digital Network (BISDN) based on the ATM network technology, which forms the core of the GII.

The BISDN/ATM network is envisioned as the ultimate network solution, because of its capability to efficiently use its resources, while integrating video, voice, data and multimedia services. The efficient use of the network resources is of paramount importance to both network providers and users, because of the explosive growth of the network traffic that we are experiencing today, and the billions of dollars that are spent annually on improving the network infrastructure. The support of many diverse services implies a very wide range of Quality of Service (QoS) guarantees, however, which usually causes a poor resource utilization. Therefore, the design of a network that efficiently shares its resources between various users with varied and often conflicting requirements is a real challenge.

The thesis approach is based on stochastic modeling. The ATM network is modeled as a queuing network, where the arrival traffic streams are represented with “appropriate” traffic models, and the network nodes are represented as queues connected with constant rate servers, corresponding to the ATM links. In order to achieve higher network efficiency, the network traffic is classified into four traffic classes: Low Loss and Delay class, Low Delay class, Low Loss class and High Loss and Delay class (LX, LD, LL and HX respectively), according to the source-specific QoS requirements. The analysis has shown that highest potential gains can be achieved by treating the LX and HX classes separately, and combining the LD and LL classes to share the same resources.

The analysis itself follows a multi-level approach: the single-link analysis considers an ATM link in isolation, whereas the network analysis considers the ATM network as a whole. The single-link analysis focuses on the so-called Low Delay Or Low Loss (LDOLL) queue, a novel priority queuing strategy which improves the performance of both LD and LL traffic classes. The performance of both single-outlet and multiple-outlet LDOLL queue is extensively analyzed under various traffic conditions. The queuing strategy achieves significant performance improvements compared to the standard First-In First-Out (FIFO) queuing, for a wide range of buffer sizes, time scales and traffic models.
Additionally the single-link analysis also examines the impact of the recently discovered self-similar traffic property on the network performance, as well as the reliability of the traditional traffic engineering techniques in the light of the new discovery. The results indicate that for practical buffer sizes the self-similarity has only a minor effect on the buffer performance. Hence the traditional engineering techniques remain applicable.

The network analysis is concerned with efficient ATM network design and in particular, with the effect of the ATM multi-layer structure and the various resource sharing strategies on the network efficiency. The main focus is on configuring a set of virtual subnetworks on top of a physical ATM network, to achieve higher utilization levels. As a result, a novel design method is developed together with the accompanying software tool, which produces a subnetwork configuration that maximizes the resource utilization for given traffic demands and QoS requirements.

Additionally the network analysis also considers the design of enterprise overlay networks on the top of an ATM network, and the impact of various design parameters on the overall design process. The obtained solution procedure and the accompanying software tool produce very good results and also illustrate the significance of the specific cost parameters for the structure of the optimal overlay network.

The thesis is concluded by a summary of the obtained results, the original contributions and recommendations for further research activities.
Chapter 1

Introduction

This thesis is concerned with the performance analysis and design of the future global information infrastructure. During the four years I worked on the subject, its importance increased considerably, due to the dramatic developments in the telecommunications, computer and media industries and the increasing dependence of society on fast and reliable communications. It is easy to see that given the huge amount of information exchanged around the world every day and the billions of dollars spent annually on networking, even a minor improvement of the efficiency of the underlying network infrastructure will produce significant payoffs. Hence there exists a genuine interest for network analysis and design.

This chapter starts with a brief description of the author's vision of the future, which emphasizes the importance of the future global network. Section 1.2 is concerned with the problems in the current networks and compares different emerging technologies as bases for the future universal network infrastructure. The rest of the chapter focuses on the ATM technology and classifies the outstanding problems related to its implementation. The last section presents the scope and outline of the thesis.

1.1 A glance at the future

In the last few years we have been witnessing several emerging trends caused by dramatic developments in the computer, telecommunications and media industries. One such trend is the increasing presence of multimedia in traditionally mono-media applications. For instance CDs were originally used only for audio storage, whereas today you can buy an interactive multimedia encyclopedia stored on a single CD. Similarly, in the early days of the World Wide Web (WWW), the information was primarily in textual form. But with the fast development of new technologies, a typical Web page today contains, in addition to the text, lots of pictures, music samples or video sequences. In a few years no one will talk about multimedia because it will have become a simple fact of life, inherent to all forms of digital communication.

Another typical trend in our society today is the increasing demand for universal and personal communications. People want to work and communicate efficiently, regardless of whether they are at home, at the office or on the road. A typical example is the
ever growing popularity of mobile telephony. In the beginning most of the users were professionals, whereas today almost everyone has a mobile phone. The ultimate goal is of course to be able to connect to a universal network from any place and at any time, and to be able to read your x-mail\(^1\), browse through your documents or contact someone in a simple and efficient way.

Certainly the most remarkable trend is the so-called Internet hype. Although the Internet itself has existed for almost twenty years, only very recently ordinary people and businesses have really started to pay attention to it. The turning point was the advent of the WWW in 1993, which started an avalanche. Suddenly everyone wanted to be connected. The general public has discovered a great new medium for exchange of information, ideas, knowledge and goods, with a very attractive economical structure. The business community has discovered an area of new business opportunities, operation structures, marketing and promotion channels, and more efficient ways for cooperation and communication. Internet has become so important today that many governments are starting to look into ways of imposing some control over it.

So, where does all this lead to? Why is the Internet so important and why are people so excited about it? The fact that Internet’s tremendous popularity is still growing in spite of the often very long response times\(^2\) indicates that the users are looking beyond the current horizon into what it might become. Actually Internet today is seen by many as the predecessor of an all-service, multimedia network of the future, which should integrate the telecommunications, computer and media (TV, radio, movie etc.) worlds into a single Global Information Infrastructure (GII)\(^3\).

The GII should provide a versatile set of services for support of both mobile and static users, and a variety of applications and media in a flexible and user-friendly manner. In other words: it will allow people throughout the world to communicate with each other using any mix of video, voice and/or data, at any time and at any place.

Another developing technology, which together with the GII will greatly influence the shape of the future world is Virtual Reality (VR). VR is an artificial environment that appears very real to its user. It allows people to see, hear or feel items and surroundings that are normally not perceptible because they are either too small or too big, or because they happened in the past or will happen in the future or just because they are imaginary. Once the technology has matured, we can expect an emergence of all kinds of distributed virtual environments (DVE), such as virtual shopping malls, virtual offices, v-schools, v-museums and similar facilities, where people\(^4\) can move around, rearrange objects and interact with each other. In principle people will be able to create any kind of fantasy environment that they can think of.

\(^1\)x-mail should be seen as combination of the today’s answering machine and the e-mail box, actually a digital post box where you can leave a message in any form: voice, video, text or even VR.
\(^2\)Another popular translation of the acronym WWW is the world wide web.
\(^3\)GII is also known as the International Information Infrastructure (III), Information SuperHighway (ISH) or the High-Performance Info-Communication Infrastructure.
\(^4\)Actually people inside an DVE are represented by a three-dimensional icons called avatars.
1.1 A glance at the future

The GII and VR technology will bring fundamental changes in the way people communicate, work, socialize, learn or entertain themselves. In other words, they will profoundly influence almost every aspect of human life, economical, political, social and cultural. In such a world of global multimedia communications and advanced VR, people will work in virtual offices together with their colleagues from around the world, as if they are all in the same building. Virtual agents would replace today's tourist agencies, insurance agents and other similar services. Artists will increasingly use a keyboard or a mouse instead of a brush or a hammer, and students will be able to follow courses at MIT, University of Cambridge and Lomonosov simultaneously.

Note two features from the preceding description that will be of great importance in the information age. The first one is the duality of the future world. The actual or real world as we know it today and a virtual, artificial world (DVEs), made out of bits traveling through network links, interwoven in such a way that it is difficult to distinguish between them. The second one is the decreasing importance of the notion of distance. On the one hand the GII will provide the means for easy communication between people who are thousands of kilometers apart and on the other hand distance-independent cost structures will make it economically feasible.

The development of these technologies will obviously have great impact on people and on society in general. Everyone will benefit. The users will get higher quality services, with increasing flexibility, mobility and comfort for both professional and spare time activities. Additionally, the availability of the multimedia communications will initiate a demand for services that we cannot even imagine today. The business world will undergo tremendous changes towards globalization and increasing efficiency. The new technology will generate new business opportunities as well as increased competition from completely new players with very different strengths and strategies. The distance-independent cost structures will have a major impact on the inter-office and telecooperative working causing globalization of labor because in principle it will be irrelevant whether the employee sits in Palo Alto, Sophia Antipolis or Bangalor.

Such developments will naturally affect society as a whole. Communication between people was traditionally limited by distance, actually by distance-based economies. Hence within an environment where distance does no longer matter, people will begin to communicate and organize themselves regardless of national borders. This will result in an emergence of many global associations of individuals and organizations, which will challenge the traditional nationwide or statewide practices of law, trade and culture, and set the foundations for a new global borderless society.

---

5Probably highly sophisticated descendants of the current software agents (see www.harvard.com and www.ac.com).

6Current communication cost structures dependent greatly on distance. However, the latest experiences with the Internet services and with the introduction of the fiber technology, show that distance is no longer a valid basis for cost; the future cost structures will probably depend on the requested service quality and bandwidth requirements.

7Whether there is any hope for reasonable relations between individuals and groups belonging to different nations, religious, races or any kind of distinguishing "marks" is out of the scope of this thesis. For further insights into these questions I refer to [Kow97] and the references therein.
But whatever shape or form the future world takes, it will be determined by the desires and needs of the people living in it. Judging by the current trends and developments, the information age has already started, and for better or for worse, it is here to stay. However, before we enter this (great?!?) new era, many technical and economical problems have to be resolved in addition to the legal and political aspects. This thesis is just a modest contribution to the technical research efforts of many researchers, professionals and enthusiasts all over the world who are trying to bring the future one step closer.

1.2 Evolution of the communication networks

The realization of the GII is a necessary precondition for the global information society of the future. From a user perspective, the GII consists of several parts (see figure 1.1):

- The public network part, managed by the telecommunications operator companies, that provides services to the general public (the telephone network is an example),

- the private network part, usually managed by an organization or university, which provides services only to organization members or students (e.g. a campus university network),

- the user equipment, typically owned by the users, which allows them to connect to the network.

From a technical and economical point of view the most intensely used part of the network infrastructure is its core, or the public part, since it provides services to everyone. Therefore there exists a great pressure to build an efficient public broadband network which will be able to handle the transport and switching of all types of services and
1.2 Evolution of the communication networks

media, using the same infrastructure. This ambitious goal is called the BISDN\(^8\) because it is considered as the logical extension of the current ISDN. It is envisioned as the core of the GII, capable of handling any mix of voice, data, video and multimedia while using its resources efficiently.

However the current situation in the telecommunications is very different. Today’s communication networks are characterized by service specialization. This means that for each type of service there is a corresponding network (e.g. telephone, data and cable TV networks). Such a situation has serious deficiencies. First of all it is a rather expensive way to provide services, because each network has to be designed, built and maintained separately, satisfying only its own service quality requirements. In addition, the service specialisation results in an inability to share resources between different networks, at least not in a efficient manner. Consequently the networks have to be dimensioned for their worst case traffic conditions.

Another typical problem in today’s telephone and cable TV networks is inflexibility, which results from their dependence on the service fixed bandwidth requirements. For instance the network is unable to gain from the advances in the coding and semiconductor technologies, which impact services and thus change their bandwidth requirements (e.g. with the current audio coding techniques voice could be compressed to bit rate lower than 10 Kbit/s, whereas the ISDN voice channel is 64 Kbit/s). Similarly the expected introduction of new services will produce a whole range of yet unknown requirements.

One of the major problems in today’s networks is the lack of capacity, and in particular the inefficient bandwidth management. The most remarkable example is the Internet. Everyone who has used the Net knows that the most frustrating thing about it is the time you spend waiting while the new web page is downloaded. The long waiting times, caused by network congestion, are getting worse all the time as the network traffic increases. Real-time communication across the Internet is often virtually impossible. The reason for this is the way bandwidth is managed within the Internet. There is actually no bandwidth management. When the network gets congested it simply drops packets, the “send-and-pray” philosophy. A common solution to the problem is to increase the bandwidth of the network trunks. This only postpones the problem, however, since Internet traffic doubles roughly each year.

Another problem characteristic for the data networks and also partly for the cable TV networks is the presence of many proprietary technologies. For instance, currently almost 70% of the Internet is driven by Cisco routers, which are running Cisco’s proprietary Internetwork Operating System (IOS) software in order to route the packets throughout the network. Since the fastest growing segment of the telecommunications market is international communications, and the equipment on the both sides of the border need not be supplied by the same vendor, the abidance to standards is a must.

The BISDN is envisioned as the network platform which should solve all of the above problems, i.e. it should be: a universal network able to transport all existing and any future services in the same way, flexible and future safe regarding the ever changing

---

\(^8\)Broadband Integrated Services Digital Network.
bandwidth and Quality of Service (QoS) requirements of different services, efficient in the use of its resources and based on worldwide standardized protocols.

The need for a universal flexible future network together with the progress in technology and system concepts, has led to the definition of a new telecommunication transport technique called the Asynchronous Transfer Mode, or ATM for short. ATM is a flexible platform able to support a wide variety of transport technologies with different framing structures (ISDN, frame relay, FDDI etc.), a wide range of services with various QoS requirements (ranging from "best-effort", like the Internet, to guaranteed upper delay bounds and tightly controlled delay variations), and efficient management of network resources. Therefore it comes as no surprise that in 1987 ITU-T chose the ATM as the transfer mode for the future BISDN.

Early ATM proponents thought ATM would be applied everywhere, from the desktop to the network core, i.e. a ubiquitous network. Such visions maybe appealing from an integration point of view, but in reality it is rather unrealistic to expect that the variety of current systems will coverage to a single technology. This can also be seen from the implementation process of the ATM technology, which has proven rather difficult, even to the extent that there are a lot of stories in today's press about the "gloom and doom" of ATM, or even its death. Such stories have led some people to believe that ATM is just another niche technology which will never really mature. This however is very far from the truth.

The difficulties that ATM is experiencing are limited to the private (enterprise) network market and they result from the differences that exist between the public and the private sectors. First of all, the bandwidth is not at all that scarce within LANs and campus networks, so it is much less important to be efficient here. Second, there is usually a single access technology, typically provided by a single vendor, so the multiplatform capabilities and the worldwide standardization of ATM technology does not deliver much of an advantage. Thus the best candidate for a ATM killer application in the enterprise environment is multimedia traffic like interactive video, voice and related applications. These applications do not tolerate delay variations or fixed delays larger than about 100 milliseconds because of the direct human involvement on the both ends. ATM with its ability to provide guaranteed QoS is currently the only technology that can satisfy such requirements. The only problem is that customers have yet to implement such applications in any significant numbers. Hence without a high-volume killer application the penetration of ATM into the enterprise depends on the traditional price/performance ratios.

On the other hand alternatives like Fast or Gigabit Ethernet are becoming faster and cheaper all the time and new queuing strategies such as weighted fair queuing are being added to the routers in order to introduce different traffic quality levels into the IP network. Since Ethernet is likely to cost one third of ATM for the same amount of bandwidth, many organizations will adopt a "throw-bandwidth-at-the-problem" strategy by deploying Gigabit Ethernet links and pray that the bandwidth utilization stays low enough to prevent important traffic from being affected. However, if the traffic demands continue to grow with the current rate and delay sensitive applications are expected to be run over the enterprise network in the future, then an ATM enterprise backbone with
1.2 Evolution of the communication networks

Ethernet on the edge makes a lot of sense as it provides an infrastructure for unforeseen new applications [Pas97].

In contrast, in public networks, ATM is very successful and it is hardly questioned whether it is the right technology. The reasons for that are fourfold. Firstly ATM supports a range of different transport technologies like ISDN, Ethernet, T1/E1 leased lines, frame relay, FDDI, DQDB, xDSL, WLL and ATM itself, allowing smooth evolution from the current legacy networks to a single multiservice infrastructure. Secondly, the public network core business asset is bandwidth, and efficient bandwidth management is a very attractive feature of ATM. Thirdly, ATM can offer a wide range of services with guaranteed QoS levels, provisioned and managed independently. And finally, ATM is a worldwide standardized technology allowing for easy interoperability between multivendor equipment, which is of paramount importance in the public arena, where most operators must interlink their networks. As a result ATM is becoming the core of many, if not all, carrier backbone networks and the trend is expected to continue.

The only visible alternative to ATM today is IP over SONET/SDH\(^8\). The concept is based on the idea that IP will be the dominant network traffic in the future, providing also different service quality levels for time-sensitive applications. The bandwidth scalability is solved through the SONET/SDH infrastructure and gigabit hardware routers should replace today's software routers, which are currently the main traffic bottlenecks.

Even if it is too early to say whether IP over SONET will be able to support applications like real-time voice and video, the existing drawbacks imply that it is an unlikely candidate for the GII backbone technology. Although the assumption that IP will be the dominant network protocol is sound, there is no clear evolution path from the current state of affairs to an all-IP infrastructure, except with a technology revolution, which is hardly likely. Bandwidth management should be done through priority schemes like weighted fair queuing or per-flow queuing and the ReSerVation Protocol (RSVP). However, prioritization is done independently by each router, which is not real traffic control. Hence the resulting service quality very much resembles the "best-effort" service quality. There are also serious problems with the RSVP's scalability, which is reflected in the latest proposals through new suggestions to use RSVP exclusively for video (see [For97]).

So the prospects are that the variety of existing and evolving networks in the telephone, data and TV/Radio worlds will converge to a single, global, multi-service future infrastructure based on the ATM technology. A common ATM core will interlink diverse access networks like ISDN, Ethernet, or cable TV networks. Because of the fundamental differences between the three worlds with respect to the nature of the traffic, service mixes and the customer needs and demands, the access networks are likely to continue to differ from each other (see [Arm97] for an in-depth discussion).

The rest of the thesis solely considers ATM networks and in particular more efficient ways of bandwidth utilization and network design. Related technical problems and unresolved issues are also discussed. The next section describes ATM functionality in detail and shows how ATM technology satisfies the requirements discussed here.

---

\(^8\)Synchronous Optical Network/Synchronous Digital Hierarchy.
1.3 Asynchronous Transfer Mode

The Asynchronous Transfer Mode represents a classic compromise, bridging the gap between circuit and packet switching. It is a form of packet switching with a virtual connection set-up phase. Before any information is sent, a virtual connection through the network is setup by a signaling procedure. The procedure also allocates the resources that are required by the connection. The user data is segmented into small fixed size packets called ATM cells, which are then sent out on the link. All cells follow the same route across the network, which allows direct control of the traffic flows throughout the network. Cells of different connections using the same link are multiplexed into a single stream, allowing for statistical resource sharing. After arrival at the destination the cells belonging to the connection are assembled and given to the appropriate application. The allocated network resources are freed after the communication has ceased.

A network based on such a transfer mode has unlimited bandwidth flexibility, i.e. the cell rate of a connection can vary freely, even throughout the duration of the connection and it can cope with very high bandwidth connections because of the reduced number of functions performed in the network nodes (no link-by-link error correction and no routing during the information transfer).

![ATM cell structure](image)

Figure 1.2: The ATM cell structure.

The ATM cell is the basic information transfer unit in the ATM network. Figure 1.2 shows the structure of an ATM cell, as defined by the ITU-T Recommendation I.361 [Tel93a]. It is 53 octets long, 5 bytes for the cell header and 48 for data. If the payload is shorter than 48 bytes, padding is used to extend the cell to full length. The header itself consists of five or six fields, depending on the interface (NNI or UNI).

The main function of the header is to route the cell from the origin to the destination point. The routing information is contained in a two-level hierarchy: a Virtual Channel Identifier (VCI) and a Virtual Path Identifier (VPI) field. The VCI field is used to
1.3 Asynchronous Transfer Mode

distinguish between various virtual channels traversing a certain ATM link. Several virtual channels following the same “path” can be bundled into a virtual path. A virtual path is identified by the VPI. Figure 1.3 shows the relationship between the physical links, virtual paths and virtual channels.

Figure 1.3: The relationship between the physical transmission path, virtual paths and virtual circuits.

The actual switching is done by means of header translation, i.e. VPI and/or VCI swapping. During the connection set-up phase, each node (ATM switch) along the route sets up an input-output pair of VCI/VPI numbers in its look-up table. When a cell arrives at the ATM switch, the switch looks at the VCI and/or the VPI numbers of the arriving cell, and uses the look-up table to determine the new values of the VCI and VPI fields. It then modifies the header, and puts the cell on the appropriate outgoing link (see figure 1.4). An ATM switch can do VP switching only or both VP and VC switching.

Figure 1.4: Switching in ATM networks

Note that the switching technique makes the VCI and VPI only locally significant, and when the connection is released the VCI and/or VPI values on the involved links are
released too, so that they can be reused by other connections. Therefore VPI and VCI fields can be kept very short (16+8(12) bits only). Another advantage is that since each ATM cell contains its own routing information, it is also the basic multiplexing unit. This enables direct multiplexing/demultiplexing of virtual channels, where each channel may have a different bit rate (the flexibility property).

The other fields have the following functions. The first four bits GFC (Generic Flow Control) are present only at the user network interface (UNI). Inside the network the GFC field is replaced with additional four bits of the VPI field. The GFC will be used to ensure fair and efficient use of the available capacity by the terminals at the edge of the network. The Payload Type Identifier, PTI, is used to distinguish between the user information cells and special network types of cells, like signaling, maintenance or idle cells. Table 1.1 gives an overview of the payload types as defined by the ITU-T Recommendation I.361 [Tel93a]. The Cell Loss Priority bit, CLP, indicates a two-level priority within a single virtual channel and it is used during network congestion conditions, when cells have to be discarded. The header information is protected by a check sum contained in the HEC (Header Error Control) field. Note that the HEC protects only the header, so the application itself is responsible for protecting the payload.

<table>
<thead>
<tr>
<th>PTI bits</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>00x</td>
<td>User data cell, congestion not experienced. ATM-user-to-ATM-user indication = x.</td>
</tr>
<tr>
<td>01x</td>
<td>User data cell, congestion experienced. ATM-user-to-ATM-user indication = x.</td>
</tr>
<tr>
<td>100</td>
<td>OAM F5 segment associated cell.</td>
</tr>
<tr>
<td>101</td>
<td>OAM F5 end-to-end associated cell.</td>
</tr>
<tr>
<td>110</td>
<td>Resource management cell.</td>
</tr>
<tr>
<td>111</td>
<td>Reserved for future use.</td>
</tr>
</tbody>
</table>

\[ F5 \text{ is the designation for the OAM flow at the virtual channel level.} \]

Table 1.1: Interpretation of the Payload Type Indicator (PTI).

ATM offers point-to-point bidirectional or point-to-multipoint unidirectional virtual connections at either VP and/or VC level. The connections can be established semi-permanently or for the duration of the call in the case of switched services. Switched connections are established through signaling procedures, whereas semi-permanent connections are established through administrative procedures. The establishment includes not only VPI and VCI allocation, but also allocation of bandwidth and provision of required QoS guarantees. For switched connections they are negotiated between the user and the network during the call set-up phase and if necessary, they are renegotiated during the call itself.

Since ATM is a connection-oriented transfer mode, bandwidth has to be reserved in the network for each virtual connection. The big advantage of ATM is its ability to statistically multiplex virtual connections with various characteristics. Statistical multiplexing means
that when multiple connections share the same link, the amount of bandwidth reserved is less than the total of the peak rates of all connections. The notion is based on the assumption that very rarely all connections will be sending at their maximal bit rate simultaneously. Therefore it suffices to reserve for each connection less than its peak bandwidth in order to achieve the required QoS\(^{10}\) (e.g. very low cell loss probability, like \(10^{-9}\)). However, the amount that can be saved depends to a great extent on the number of multiplexed connections, their traffic characteristics, their interdependence and the QoS they require.

1.3.1 ATM Protocol reference model

The ITU-T Recommendations on BISDN relate to two interfaces: the User-to-Network Interface (UNI), describing the interface between a user terminal and the network, and the Network-to-Network Interface (NNI), describing the interface between the network nodes. While BISDN is a definition for public networks, ATM can also be used within private networking environments. In recognition of this fact, ATM Forum has defined two distinct forms of ATM UNIs [ATM93](see figure 1.5):

Public UNI – for interconnecting ATM users\(^ {11} \) with an ATM switch deployed within the public service provider’s network,

Private UNI – for interconnecting ATM users with an ATM switch which belongs to the same corporate network.

The primary distinction between the two UNIs is physical reach. Facilities at the Public UNI must be capable of spanning long distances, whereas private switching equipment can also incorporate limited-distance technologies.

The ATM protocol reference model defined in Recommendation I.121 follows the same logical hierarchy principles as the famous OSI model. The model shown in figure 1.6 is divided into multiple (vertical) planes and (horizontal) layers. The planes are used to distinguish between user, control and management functions:

U-plane: The user plane provides transfer of user application information.

C-plane: The Control plane deals with connection establishment and release and other related functions necessary for providing switched services.

M-plane: The Management plane provides layer management functions and the capability to exchange information between the U-plane and the C-plane.

The layers divide the protocols into independent sets. The model distinguishes between the physical layer (PHY), the ATM layer, the ATM adaptation layer (AAL) and higher-layer protocols (HLP). The protocol layers involved at both UNIs are limited to

---

\(^{10}\)In general reserving less bandwidth yields lower service quality, i.e. higher loss rates, larger delays and delay variations.

\(^{11}\)An ATM user is defined as any device that makes use of an ATM network through the Public UNI. Hence both an ATM enabled device and a privately owned ATM network can be abstracted as an ATM user.
the physical and ATM layer only and some C-plane higher-layer protocols for support of switched virtual connections. These two layers are service independent and contain functions applicable to all upper layer protocols, i.e. they are independent of user applications (see figure 1.5). So in the U-plane all service-dependent functions (AAL and higher-layer protocols) operate end-to-end, which makes the network core service independent. As a
1.3 Asynchronous Transfer Mode

consequence, adding new services is very simple because it only requires changes in the user terminals.

The physical layer is composed of two sublayers: the physical medium (PM) sublayer, which supports pure medium-dependent functions and is responsible for correct transmission of bits across the physical medium, and the transmission convergence (TC) sublayer, which provides timing information and converts the ATM cells into a bit stream that is transported over the physical medium. Possible transmission media are SDH/SONET, PDH, FDDI or cell-based transmission, with speeds ranging from 1.5 to 622 Mbit/s and higher.

The ATM layer is entirely independent of the physical medium used to transport the ATM cells. Therefore it can interconnect different technologies (multitechnology platform). It performs the following functions: multiplexing and demultiplexing of cells of different connections (identified through their VCI/VPI values) into/from a single-cell stream on a physical layer; VCI and/or VPI translation when switching a cell from one physical link to another; provision of appropriate QoS to distinct virtual connections as well as within a single virtual channel (CLP bit); management functions like congestion indications and ATM user-to-user indication and addition/extraction of the cell header after/before the cell is delivered from/to the AAL.

The AAL isolates the higher layers from the specific characteristics of the ATM layer by mapping the higher-layer protocol data units (PDU) onto the information field of the ATM cell and vice versa. It performs service-dependent functions for all three planes.

In order to minimize the number of AAL protocols, the following service categorization has been defined by the ATM Forum12 [ATM96]:

**CBR** — Constant Bit Rate service category, used by applications that require a static amount of bandwidth (i.e. constant transmission rate), continuously available during the connection lifetime and stringent QoS requirements (real-time applications).

**rt-VBR** — Real-Time Variable Bit Rate service category, intended for real-time applications with specified traffic parameters, transmitting at variable rate in time (bursty applications), allowing therefore for statistical multiplexing savings.

**nrt-VBR** — Non-Real-Time VBR service category, intended for bursty applications with specified traffic parameters, but without the stringent delay requirements (non-real-time applications).

**ABR** — Available Bit Rate service category, intended for bursty applications that can adapt their transmission rate according to the changing conditions inside the ATM network (available bandwidth).

**UBR** — Unspecified Bit Rate service category, intended for non-real-time applications without any specified traffic parameters and QoS requirements.

---

12ITU-T has defined a slightly different classification and nomenclature (see Recommendation I.371 [Tel93d]).
Note that each service category can be used by a multitude of user applications with varying QoS requirements.

Currently four AAL protocols are recommended by the ITU-T, namely AAL1, AAL2, AAL3/4 and AAL5 (for definitions see Recommendation I.363 [Tel93b]). Possible combinations of certain AAL protocols with some of the service categories discussed above are described in Recommendation I.363 [Tel93b], but specific associations are for further study.

1.4 Open issues in ATM networks

The previous section considered the functional details of the ATM network that are already standardized. This section, however, is concerned with technical problems and ATM implementation issues that are still under discussion. They can be classified in two generic classes: network design and traffic control related problems. Network design is concerned with the minimal cost network architecture that satisfies the QoS requirements of all connections. The traffic control or traffic management, on the other hand, is concerned with maintaining the requested service quality levels when the network is in use. This requires development of effective mechanisms that can control the traffic offered to the network. Note that while traffic control functions operate in real time, network design actions have much more relaxed time constraints.

Before discussing the network design and traffic control issues in ATM networks, the section defines a time scale framework, which is used throughout the thesis to structure the discussion about various ATM problems. Many authors have proposed some kind of multi-level ATM framework [Sch88], [HGM91], [FHW96], where each level captures traffic behavior on another time scale. This thesis follows a combined approach of [Sch88] and [FHW96] with some adaptations, where the higher two levels are typically used to categorize network design activities, while the lower two levels are used to categorize traffic control activities, and the middle level is used in both types of activities. The traffic activity at each level is conditioned on the activity of the next higher level (see figure 1.7).

The longest time scale - the PHYSICAL level - is characteristic for general long-term design planning. The long-term design actions typically operate on the physical network architecture. The events correspond to physical network upgrades due to major changes in traffic patterns. The inter-event times are roughly in order of months.

On a somewhat shorter time scale we find the PATH level. The design actions at this level operate on the virtual path overlay network. This level actually comprises two sublevels: the global PATH sublevel and the local PATH sublevel. The global PATH sublevel algorithms perform network-wide modifications to the VP topology and capacity assignments, and are triggered by traffic changes based on the time of day or by major network link failures. The inter-event times are roughly in the order of a part of a day periods. The local PATH sublevel algorithms on the other hand include strategies for dynamically making only minor changes to the VP topology and capacity assignments. The events coincide with small variations in network traffic and minor failures, and the inter-event time is roughly in the order of hours.
The lower three levels correspond to the CONNECTION level, BURST level and CELL level, respectively. They are best described by considering the behaviour of a single traffic source. Events at the CONNECTION level correspond to establishment and tear-down of virtual connections, and the inter-event times are roughly in order of minutes.

During the lifetime of a virtual connection, the source activity ranges from inactive (silent) to transmitting at its peak rate. The time scale at which the source alternates between different states of activity is called the BURST level. The events correspond to the source rate changes, and the inter-event times are in order of milliseconds.

The fastest time scale is the CELL level. The events correspond to a transmission of a cell at the rate of the ATM link and inter-event times are usually in order of microseconds.

### 1.4.1 Network design

Network design algorithms at the PHYSICAL and PATH levels are conceptually very similar; the differences amount to various constraints on the variables of the network model and the available execution time for the algorithm. The algorithms have basically two stages, which are performed in an iterative fashion. In the first stage the network topology has to be determined, i.e. location of nodes and/or interconnecting links is resolved. The second stage is the dimensioning phase, which allocates network resources given the network topology. This involves dimensioning of link capacities and buffers, in order to satisfy the QoS constraints. Naturally the optimization process tries to minimize the network cost for given traffic loading and QoS requirements.

Many design algorithms have been developed in the last two decades, both for circuit-switching [Kel86, Kel88, Kel90, Gir90] and packet-switching networks [GK77, BF77, MS86, Gav86, FA92, Ker93]. Since ATM is connection-oriented, circuit-switching algorithms are considered more suitable for ATM network design. They provide a very convenient way of network analysis by considering each network link separately, which in turn greatly simplifies the performance analysis.

Design of ATM networks however, differs considerably from design of circuit-switching networks. First of all, ATM networks will integrate many distinct services with a variety of
QoS requirements. This results in a very complex relationship between the offered traffic and the reserved bandwidth necessary to satisfy those requirements, which is essential to the efficient utilization of the network resources.

Another difference that further complicates the design process is the multilayer structure of the ATM network. As already described in section 1.3.1, the ATM network is composed of three layers: a physical, a virtual path (VP) and a virtual connection (VC) layer. The interdependence between them is illustrated in figure 1.3. The complexity stems from the fact that the network topology at each layer needs not to be the same, but the link dimensioning at all three layers is interdependent.

Additionally, the complexity of the design procedure is also dependent on the level of statistical multiplexing applied in the network. For instance, the VP capacity can be dedicated to a single VC, it can be shared between VCs in the same service class or it can be shared between VCs of all service classes. The problem is that a greater extent of resource sharing does not automatically translate into higher resource utilization.

Therefore the ATM network design requires extension of the current circuit-switching methods as well as development of new design algorithms. The thesis describes a novel design method that incorporates all relevant issues.

1.4.2 Traffic Control

ATM traffic control is fundamentally related to the ability of the network to provide differentiated QoS for network users. Its objective is to protect the network and its users from congestion in order to achieve the agreed level of quality, and in the same time to realize high utilization of network resources. There is of course a trade-off between these two goals; better service quality implies lower resource utilization and vice versa. Therefore in order to achieve both targets, increasingly complex control procedures are necessary.

There are several properties of the ATM technology that make traffic control more difficult. For instance, the support of a variety of service types with very heterogeneous quality requirements. Additionally real-time services are much less controllable than data services\textsuperscript{13}. Also the very high transmission speeds result in a high bandwidth-delay product, which makes reactive control techniques less suitable than preventive control techniques.

Traffic control in ATM networks has been the subject of intense studies in recent years, which resulted in definition of a rich set of various traffic control functions, some of which are already standardized (see [ATM96]). However, there are no final recommendations with respect to the assigning control functions to service categories, and no limitations for including new control functions in the standards. For an in-depth discussion of all options and considerations and an extensive list of references, see [RMV96].

In general ATM traffic control functions can be subdivided according to the time scale at which control actions occur: CONNECTION level, BURST level or CELL level, respectively.

\textsuperscript{13}There is no point in delaying the cells of a video application, because if they arrive too late, they will be discarded.
The main control function that operates at the connection level is the Connection Acceptance Control (CAC). CAC is defined as a set of actions performed by the network during the connection establishment phase to determine whether a new connection can be admitted, based on the traffic parameters and the requested QoS of the new connection and the current state of the network. The CAC looks for a route through the network which can accommodate the new connection without deteriorating the service quality of the ongoing calls. If an agreement about the offered traffic with appropriate QoS between the user and the network can be negotiated, the terms of the agreement are drawn in a contract. The network guarantees the agreed QoS as long as the user adheres to its traffic description\(^\text{14}\).

The burst level control is adequate for sources with known traffic characteristics and peak rates substantially lower than the speed of the link, and for sources that can dynamically adjust their transmission rates. Typical examples of burst level control functions are traffic shaping and ABR flow control. Traffic shaping mechanisms are used to modify the user traffic characteristics in such a way that they comply with the specified parameters in the traffic contract. The ABR flow control may be used to adaptively share the available bandwidth among many users in a fair way. It continuously informs the traffic source about the network congestion state (i.e. the amount of bandwidth still available), so that the source itself can decide when to transmit its next burst and at what rate.

Cell level control functions operate on individual ATM cells. In general they describe different strategies for storing, retrieving or discarding cells from ATM switch buffers. Typical examples are usage parameter control (UPC) and various priority queuing schemes. UPC is used in relation to the CAC. Its function is to police whether the users abide to the traffic parameters specified in the contract. Nonconforming cells are discarded or tagged with the CLP bit, so that if congestion is encountered inside the network, they will be dropped first. The need for priority queuing comes from the diversity of the QoS requirements that ATM network has to provide. Many different solutions have been proposed (see [KHBG91, JS96, RMV96] for an overview) but none of these has been standardized yet. This thesis analyzes a novel priority queuing mechanism that can significantly improve the network resource utilization.

### 1.5 ATM performance modeling

As discussed in the previous section, to achieve the desired QoS at the lowest possible cost, or i.e. to build an efficient broadband network, one has to evaluate the performance of the ATM network under various traffic conditions. In other words one has to perform ATM performance analysis. The general approach is to develop an accurate model for both the traffic and the network, which will be simple enough to allow mathematical analysis or at least simulation, and at the same time it can capture the relevant network congestion characteristics.

\(^{14}\)Note that CAC mechanisms are also a part of the network design algorithms, because they play an important role in determining the routes of the virtual connections in the ATM network.
Traffic modeling aims to describe the statistical properties of the traffic offered to the network. It has been a popular area of research and there exists an immense number of traffic models, the majority of which are based on Markovian processes (see [Fro94] for an overview). However, in the last few years detailed studies of traffic measurements obtained from different operating networks: Ethernet LANs, WWW, VBR video, ISDN, CCSN/SS7 ([LTTW94, LTTW93, BSTW95], [CB96], [GW94], [EW94], [DMRW94]); have produced several remarkable results:

- The actual network traffic is **statistically self-similar** (see section 2.2 for definitions), i.e. the traffic bursts are observed at all time scales, from milliseconds to hours,

- aggregating distinct traffic streams typically **intensifies** the burstiness (self-similarity), which is in sharp contrast to the conventional teletraffic wisdom based on the folklore theorem\(^{15}\).

This fractal behavior of the network traffic is statistically very different from the traffic generated with any of the currently used traffic models. This has caused a lot of concern in the research community, especially after initial results suggested that the performance of queuing models under traffic generated with self-similar traffic models is much worse than the predictions obtained with traditional traffic models suggested [ENW96, LTG95, Nor94]. However, aside from its ubiquitous presence in all of the measured traces and its statistical significance, the practical relevance of the self-similarity is determined by its impact on the network performance.

Network modeling also tries to describe the network itself, including all functional details that could impact its performance. The ATM network can be seen as a set of nodes operating in store-forward regime, connected by ATM links. The traffic consists of ATM cells following their routes across the network. At each node along the route the cells are 'processed' and then sent on to the next ATM link. Thus ATM networks can be modeled as queuing networks, where network nodes are represented as queues with constant rate servers, and the statistically multiplexed cell arrival streams are modeled with "appropriate" traffic models. By analyzing a queuing network, one is able to calculate the relevant performance measures for a given network topology and traffic description, which in turn enables one to express the network performance in terms of the network design parameters. In other words, one is able to perform proper network design. Similarly, traffic control mechanisms can also be represented by queuing models and are thus amenable to the same analysis approach.

### 1.6 Scope and Outline of the thesis

The thesis considers performance analysis and design of ATM networks. It is restricted to ATM networks only. Other broadband networking solutions are not considered for the reasons given in section 1.2. The objective is to develop various mechanisms/methods for increasing the efficiency of the ATM network.

---

\(^{15}\)The folklore theorem basically says: **multiplexing a large number of independent traffic streams results in a Poisson-like (smoother) traffic.**
1.6 Scope and Outline of the thesis

Traffic control considerations are restricted to CELL level mechanisms, focusing on the local buffer priority control and the impact of the BURST level variations on the buffer performance. Both per-flow buffering and shared buffering are analyzed.

The performance analysis of the self-similar phenomenon considers both FIFO and priority queuing strategies. The analysis is restricted to the CELL and BURST level. CONNECTION level performance analysis is outside the scope of the thesis.

The thesis considers network design algorithms at the global PATH sublevel only. The local PATH sublevel and the PHYSICAL level are outside the scope of the thesis. The design procedures account for design of the VP and VC layers and examine different statistical multiplexing strategies (complete sharing, partial sharing and complete partitioning).

The thesis can roughly be divided into two parts: performance analysis (Chapters 2 and 3), and network design (Chapters 4 and 5). The performance analysis part focuses on the BURST and CELL level performance (cell loss probability and cell delay) of a single ATM link. It investigates to what extent the network resource utilization can be improved when complementary services are statistically multiplexed on a single link. It also investigates the relevance of the self-similarity property for the network performance analysis. In the part on the network design the emphasis shifts to CONNECTION and PATH level measures and other ATM network design issues. At the CONNECTION and PATH level one is interested in the actual savings in costs that can be achieved by implementing different resource-sharing strategies and the impact of the interdependence of the ATM layers on the network structure.

Chapter 2 is concerned with network traffic characterization and modeling. Section 2.2 provides the mathematical framework for the treatment of the self-similar phenomenon in the network traffic. Section 2.3 considers various traffic models at the lower three levels, traditional as well as self-similar, and investigates their quality compared to measured traffic patterns.

Chapter 3 is concerned with the CELL level and BURST level analysis of a single ATM switching buffer. Section 3.1.2 presents an ATM multiplexer, which improves the performance of both loss-sensitive and delay-sensitive cell flows by assigning space and time priorities to the respective cell flows. Section 3.1.3 extends the approach by finding optimal queuing strategies when multiple links are sharing the same buffer. Section 3.2 focuses on the on the BURST level performance of ATM switches with and without priorities. The last section examines the relevance of the self-similarity property, by comparing the performance of an ATM switch loaded with both measured and synthetic traffic traces.

Chapter 4 is concerned with ATM network design. It discusses relevant design aspects, and in particular the impact of the ATM multilayer structure on the network efficiency. The focus is on finding an optimal VP subnetwork configuration on top of the physical network that maximizes the efficiency of the ATM network. Since the formulated design problem is very complex, section 4.3 describes a heuristic solution approach based on a generic design framework that iteratively solves a sequence of relevant subproblems. Section 4.4 describes BANDIT, a software tool that implements the design framework. Finally section 4.5 investigates the quality of the design framework by analyzing the preliminary experimental results.

Chapter 5 is concerned with a design of a connectionless ATM overlay network, as a
special case of the complete partitioning resource sharing strategy. In addition the design process is considered from the network user viewpoint rather than the viewpoint of the network provider. The chapter considers the trade-off between the network node costs and link capacity costs, and between the cost of network elements (nodes and links) and performance measures (network delay).

Chapter 6 concludes the thesis by summarizing the obtained results and their limitations as well as topics for further research.

The original contributions and the new results presented in the thesis are listed in Table 1.2.

<table>
<thead>
<tr>
<th>Contribution</th>
<th>Description</th>
<th>Reference</th>
<th>Part</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traffic model verification</td>
<td>Validation of the quality of both traditional and self-similar traffic models</td>
<td>[Har97], [PSS96]</td>
<td>2.3.2</td>
</tr>
<tr>
<td>Multi-outlet LDOLL queue</td>
<td>Generalization of the LDOLL discipline to shared buffer with multiple outlets</td>
<td>[Sta94], [SSA93]</td>
<td>3.1.3</td>
</tr>
<tr>
<td>LDOLL BURST level analysis</td>
<td>Performance analysis of the LDOLL queue for bursty traffic and larger buffer sizes</td>
<td>[Har97]</td>
<td>3.2.2</td>
</tr>
<tr>
<td>Relevance of the self-similarity</td>
<td>The impact of the self-similar property on the buffer performance</td>
<td>[Har97]</td>
<td>3.2.3</td>
</tr>
<tr>
<td>ATM network design framework</td>
<td>Optimal design of VP subnetwork configurations</td>
<td>[KS98], [Klo98]</td>
<td>4.3</td>
</tr>
<tr>
<td>BANDIT</td>
<td>A flexible ATM network design tool</td>
<td>[KS98], [Klo98]</td>
<td>4.4</td>
</tr>
<tr>
<td>B-NET</td>
<td>Optimal design of connection-less enterprise ATM overlay networks</td>
<td>[SdJvLitV95], [SS95]</td>
<td>5.3</td>
</tr>
</tbody>
</table>

Table 1.2: Original contributions of the research described in this thesis.
Chapter 2

Characterization of broadband traffic

The first step towards analysis and design of ATM networks is to clearly specify the description of the network traffic and the performance requirements of the supported applications. A complete specification is, however, rather problematic for two reasons. First, most of the applications adapt to the facilities provided by the network, e.g. classical- and IP-telephony have similar QoS requirements but very different traffic patterns. Second, it is debatable whether it makes sense to design tomorrow's network using the requirements of the current applications; in the future the bulk of the network traffic will be due to the new and radically different applications. The approach taken in this chapter assumes that applications possess certain generic inherent properties that result from human interactions, which are therefore independent of the network infrastructure and are not likely to change in the future.

The next section discusses a number of the inherent application properties which are likely to have a significant impact on the future broadband network, keeping in mind that current applications will continue to play a role in the future networks, especially in the beginning. Section 2.2 is concerned with the self-similar property of the network traffic. Apart from the formal definitions, it also discusses a possible physical explanation of the omnipresence of the self-similarity in the network traffic traces. Section 2.3 is concerned with modeling of the network traffic. It describes different traffic models, which are used throughout the thesis, classified according to the activity level they model. The emphasis is on theburst level models, in particular on traditional models, based on Markovian assumptions, and self-similar models, based on processes that exhibit self-similar behavior. Additionally the section presents measured traffic traces together with their statistics. The traces are used as reference traffic in order to determine the validity of the presented theoretical models.

2.1 Broadband traffic classification

From a traffic management point of view the emergence of modern high-speed networks and the prevailing trend towards BISDN combines drastically new and different transmission and switching technologies with a very heterogeneous mixture of services and
Table 2.1: The application matrix contains different example applications belonging to the corresponding application classes. The abbreviations VOD and MOD stand for video and music on demand.

Consider the applications in Table 2.1. Each field in the matrix can be seen as an application class at a certain level of abstraction, corresponding to what the user is doing. The classification is derived by considering video, voice and data/image in interactive, messaging, distribution and retrieval mode. Every class contains several applications. Some typical examples are given in the table. One can also distinguish a class of computer communication applications such as LAN interconnection, distributed computing, client/server applications etc., which are not included in Table 2.1. The list is by no means complete, but it is representative and sufficient for our purpose, namely to identify the various QoS requirements that the network has to satisfy.

Let us look at the interactive audio and video communications first. They have very stringent delay and delay variation (jitter) requirements. This is because when people are interacting in real time, a lack of continuity caused by excessive jitter or large delay beyond some threshold of a few hundred milliseconds has very significant impact on the perceived quality of communication. On the other hand, audio and video can be quite tolerant to losses if they are coded properly. Loss of the header information of a video frame can cause significant quality damage\(^1\), whereas loss of any of the picture elements could easily go unnoticed, e.g. by employing error-concealment techniques [LEZ94]. Interactive audio and video applications produce a continuous flow of cells in time, which is why they are also called rate-oriented applications [ATM96].

Audio and video for distribution and retrieval applications however, are rather tolerant to delay and delay jitter\(^2\). The user is sensitive to delays of an order of a few seconds, which is much longer than the interactive delay threshold. The loss requirements are also different. Distribution entertainment video, for example, requires much smaller losses than videoconferencing.

Traditional computer communication applications are generally very sensitive to loss because all data has to be received correctly. In an ATM environment every lost cell re-

\(^1\)Usually the frame header is specially protected, for example by forward error correction (FEC) codes.

\(^2\)The jitter should be bounded though, because of the constrains on the size of the playback buffer.
2.1 Broadband traffic classification

results in a retransmission of the whole IP packet, which causes excessive delays, increases the network load and has serious impact on the network throughput [RF94]. Computer communications, however, are much more tolerant to delay and jitter. This is because computer communication applications and audio/video/image/data messaging applications are inherently unit-oriented applications and do not have a natural rate. In other words, the session is completed when all data is received. The user does not care about the time relation of the arriving cells, but he is interested in the transferred object as a whole.

Obviously in order to be able to provide the appropriate QoS and traffic control over this vast space of application requirements, some structuring of the problem is necessary. Using the loss and delay tolerance of the application as parameters, we can split the space of application requirements into four subspaces or traffic classes, as given in figure 2.1. The mapping of the applications from Table 2.1 to the traffic matrix of figure 2.1 is given in figure 2.2. The most demanding class, LX, like high quality multimedia conferencing

```
<table>
<thead>
<tr>
<th>High</th>
<th>Low Delay Tolerate Loss (LD)</th>
<th>Tolerate Loss &amp; Delay (HX)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loss</td>
<td>Low Loss</td>
<td>Low Loss</td>
</tr>
<tr>
<td></td>
<td>Low Delay</td>
<td>Tolerate Delay</td>
</tr>
<tr>
<td></td>
<td>(LX)</td>
<td>(LL)</td>
</tr>
<tr>
<td>Delay</td>
<td>High</td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 2.1: The traffic class matrix

or bank transactions, is best handled by giving it reserved peak bandwidth\(^3\), protected from the rest, and handled separately in that sense (as suggested in [New94, JS96]). This corresponds to using a CBR service, as defined by the ATM Forum\(^4\) [ATM96], i.e. it means assigning a simple, reliable, guaranteed channel to the user application. The non-demanding class HX can be assigned the lowest priority and used to fill up any spare capacity in the network, which can be identified as the Unspecified bit rate (UBR) service category [ATM96]. This class does not request any explicit guarantees from the network, hence its name. Note also that this service model, commonly known as best effort, is the basic operation mode of the current Internet.

Because the stringent QoS requirements of the LX class prevent any substantial efficiency gains, and because the HX class uses only the spare network resources, the two remaining classes in the diagonal of the traffic matrix in figure 2.1, the Low Delay (LD) and

---

\(^3\)Because of the stringent QoS requirements, there is not much to gain from statistical multiplexing of such sources, hence the peak bandwidth allocation.

\(^4\)Or the Deterministic Bit Rate transfer capability according to the ITU-T nomenclature.
Chapter 2. Characterization of broadband traffic

Figure 2.2: Mapping of the applications to the traffic matrix

the Low Loss (LL) class, have the biggest potential for achieving higher efficiency through statistical multiplexing. The LD traffic class is characterized by its sensitivity to delay and delay variance and relative insensitivity to losses, whereas the LL traffic class has complementary requirements. Video conferencing, distributed teleteaching, telephone, radio are all examples of LD applications, whereas Video on Demand (VOD), multimedia e-mail, LAN interconnection all belong to the LL class applications.

Note also that several applications lie on the border between different traffic classes. The choice of the appropriate network service obviously depends on the desired quality. For instance, if the picture quality of the VOD is not satisfactory, one may consider a more reliable and more expensive LX type of service\(^5\).

The previous discussion is only loosely connected to the ATM Forum's service categories [ATM96] or IETF's Internet service classes [RMV96]. The most appropriate mapping of applications to the specific service classes is still an open issue with many different and often contrasting views. This approach is only one of the possible realizations and its viability will be analyzed and discussed in the chapters that follow.

### 2.2 The nature of broadband traffic

The previous section discussed the properties and the QoS requirements of various current and future applications. This section focuses on the nature of the traffic that is produced by many of those applications and subsequently aggregated inside the network. More specific, it discusses a feature that was found to be ubiquitous in traffic traces from many different types of networks (Ethernet LANs, WWW, VBR video, CCSN/SS7, ISDN, WAN). As indicated in Chapter 1, thorough examination of these data sets revealed a very surprising result: the network traffic, regardless of the network type under study or the time or place of measurement, is statistically self-similar or fractal by nature. In

\(^5\)Although the ATM tariffing is still an unsolved issue I assume here that the cost will certainly depend on the quality of the requested connection.
2.2 The nature of broadband traffic

other words, realistic network traffic looks the same (similar) when measured over time scales ranging from milliseconds to hours.

Fractal is a word coined by Benoit Mandelbrot and it denotes a mathematical object whose appearance remains unchanged regardless of the distance from which it is viewed. Fractals are inherent in many patterns of nature, such as clouds, mountains and coastlines [Man83]. For example, the shape of a mountain looks like that of the hills that comprise the mountain, the shape of the tree resembles that of the branches forming the tree. More formally, a fractal object possesses a form of self-similarity, i.e. parts of the whole fit to the whole when properly scaled (see figure 2.3). Hence self-similarity and scaling are closely related.

The previous paragraph considers deterministic self-similarity. In the context of stochastic processes, self-similarity is defined in the terms of a distribution of the stochastic process. Let \( Y_t \) be a continuous time stochastic process. \( Y_t \) is called self-similar, with self-similarity parameter \( H \), if for any positive stretching factor \( c \) the rescaled process \( c^{-H}Y_{ct} \), with time scale \( ct \) has identical finite-dimensional distribution as the original process \( Y_t \) [Ber94]. This means that for any sequence of points \( t_1, \ldots, t_n \) and a positive constant \( c \), \( (Y_{t_1}, \ldots, Y_{t_n}) \) has the same distribution as \( c^{-H}(Y_{ct_1}, \ldots, Y_{ct_n}) \). Thus typical realizations of a self-similar process look qualitatively the same irrespective of the distance from which we look at them. Contrary to the deterministic self-similarity it is not one picture or shape that repeats itself, but it is rather the general impression that remains the same.

Similarly a stochastic point process is called self-similar or a fractal process when a number of relevant statistics exhibit scaling with the related scaling exponent (see also section 2.2.1). Anyway it is very intuitive to view the network traffic as a realization of a stochastic point process; each cell arrival is associated with an arrival epoch.

Assume that the previously defined continuous time self-similar process \( Y_t \) has the property of stationary increments, i.e. assume that for any \( k \) and any choice of sampling instants \( t_1 < t_2 < \ldots < t_k \), the random variables \( Y_{t_2} - Y_{t_1}, Y_{t_3} - Y_{t_2}, \ldots, Y_{t_k} - Y_{t_{k-1}} \),
are independently distributed\(^6\) [Gar94]. Let us construct a stationary increment process \(X = X_1, X_2, \ldots\), defined as:

\[
X_n = Y_{nT} - Y_{(n-1)T} \quad \text{for} \quad T > 0
\]  

(2.1)

Then, the discrete time increment process \(X\) will exhibit fractal behavior such as \textit{long-range dependence} and \textit{slowly decaying variance}\(^7\) [Ber94]. Hence the self-similar process \(Y_t\), with stationary increments, serves as an underlying process yielding a fractal process. If we assume that \(Y_t\) denotes the number of cells arrived up to a time \(t\), then \(X_n\) represents the number of cells or bytes that arrived during the \(n\)-th time interval \(T\). Hence the fractal processes are very attractive for traffic modeling purposes. Consequently the rest of this section focuses on discrete time self-similar processes and their properties.

### 2.2.1 Definitions and properties of stochastic self-similar processes

This subsection considers four related properties of discrete time self-similar processes: second-order self-similarity, long-range dependence, slowly decaying variance and the heavy-tailed property. Since there exist somewhat conflicting definitions in the literature, this subsection clearly states the basic definitions, and the related properties. It follows the approach of Tsybakov and Georgakis [TG97a]. It considers only discrete time processes.

**Exactly second-order self-similar processes**

Let \(X = \{X_n : n = 0, 1, 2 \ldots\}\) be a wide-sense stationary (sometimes called a covariance stationary) process in the discrete time domain; i.e. a process with a constant mean \(\mu = E[X_n] < \infty\), variance \(\sigma^2 = E[(X_n - \mu)^2] < \infty\) and a normalized autocorrelation function\(^8\):

\[
r(k) = \frac{E[(X_n - \mu)(X_{n+k} - \mu)]}{\sigma^2},
\]  

(2.2)

that depends only on \(k \in \mathbb{N}\).

The process \(X\) is called \textit{exactly second-order self-similar}, with parameter \(H = 1 - \beta/2\), \(0 < \beta < 1\) if its autocorrelation function is given by [Ber94, TG97b]:

\[
r(k) = \frac{1}{2}[(k + 1)^{2-\beta} - 2k^{2-\beta} + (k - 1)^{2-\beta}],
\]  

(2.3)

for \(k \in \mathbb{N}_1\). The parameter \(H\) is called the \textit{Hurst parameter}, for historical reasons [MW69], and it completely characterizes equation (2.3).

---

\(^6\)A more general definition is given in [Ber94].

\(^7\)The formal definitions of these properties are given in the subsequent subsections.

\(^8\)Strictly speaking equation (2.2) defines the correlation coefficient [Gar94]. The autocorrelation function is equal to the correlation coefficient only for a zero mean process with a unit variance. Here we are using the usual terminology found in the literature.
2.2 The nature of broadband traffic

As shown in [TG97a], the exactly second-order self-similar processes exhibit several properties: the aggregation property, the long-range dependence property and the slowly decaying variance property.

An alternative definition of the second-order self-similarity that is common in the literature is related to the following aggregation property. Given the stationary process $X$, let's construct an aggregated wide-sense stationary process $X^{(m)} = \{X_n^{(m)} : n \geq 0\}$, by averaging the $X_n$s over adjacent, non-overlapping blocks of size $m$, according to:

$$X_n^{(m)} = \frac{1}{m} \sum_{i=(n-1)m+1}^{nm} X_i$$

Then if $X$ is exactly second-order self-similar, the autocorrelation function of the aggregated process $X^{(m)}$, denoted by $r^{(m)}(k)$, is equal to the autocorrelation of the original process $X$ [TG97a, LTWW93, TG97b, LTG95]:

$$r^{(m)}(k) = r(k) \quad \text{for } k \geq 0 \quad (2.4)$$

where $r(k)$ is defined with (2.3). Non-self-similar processes, like the Poisson process or Markovian processes, on the other hand tend to second-order white noise when aggregated, i.e., $r^{(m)}(k) \to 0$ as $m \to \infty$, for all $k \geq 0$.

Another very useful property of the second-order self-similar processes is the long-range dependence (LRD) property. Namely, given (2.3), it is easy to show that

$$\lim_{k \to \infty} \frac{r(k)}{k^{-\beta}} = \frac{1}{2}(2 - \beta)(1 - \beta) = H(2H - 1) \quad (2.5)$$

where $H = 1 - \beta/2$ and $0 < \beta < 1$. Note that (2.5) indicates that $r(k)$ decays hyperbolically in the lag $k$, which implies a non-summable autocorrelation function ($\sum_{k=0}^{\infty} r(k) \to \infty$). Stochastic processes used in traditional traffic models, however, are typically characterized with exponentially decaying $r(k)$, i.e. $\sum_{k=0}^{\infty} r(k) < \infty$, hence they are also called short-range dependent (SRD) processes.

A well-known result, taught in any basic course of statistics, is that for independent observations, the variance of the sample mean is equal to the variance of one observation divided by the sample size. More formally this is expressed as:

$$\text{Var}\left[X^{(m)}\right] = \sigma^2 m^{-1} = \frac{\text{Var}[X_i]}{m} \quad (2.6)$$

Most of the time series used in the literature exhibit this behavior. However, the variance of $X^{(m)}$ of virtually all recently measured network traffic data sets does not appear to follow equation 2.6, but instead it tends to zero slower than $m^{-1}$. This is known as the slowly decaying variance property. The reason for such behavior lies in the self-similar nature of the network traffic. Namely, as proven in [TG97a, Ber94], the exactly second-order self-similar process $X$ satisfies (2.3), if and only if

$$\text{Var}\left[X^{(m)}\right] = \sigma m^{-\beta} \quad \text{for } \forall m \in \mathbb{N}_2 \quad (2.7)$$
where $\beta = 2 - 2H$. For short-range-dependent processes $H = 0.5$ and equation (2.7) reduces to the standard form of equation (2.6). Hence (2.7) can be viewed as a generalization of (2.6) which incorporates both short- and long-range-dependent processes.

An important application of the slowly decaying variance property is that it can easily be detected by plotting the $\text{Var} [X^{(m)}]$ against $m$ on a log-log plot, known in the literature as the variance-time plot [Ber94, LTWW94]. If it forms a straight line with an absolute slope less than $-1$ for a wide range of $m$, then we say that the process $X$ possesses slowly decaying variance or, equivalently, that $X$ is self-similar. The method is employed in section 2.3.2, to investigate whether the measured traffic traces possess self-similar properties.

One should also note that in reality, because of the finite length of the measured data sets, it can never be proven with certainty that $\beta \neq 1$ in equation 2.7. For a given $m$ there is always a constant $c$ such that $\text{Var} [X^{(m)}] = cm^{-1}$. If however the process $X$ is indeed self-similar, then $c$ is not a constant but it increases with $m$. Therefore, when fitting a self-similar process with an short-range-dependent model, increasing of the sample space typically leads to models with prohibitively large number of parameters.

**Asymptotically second-order self-similar processes**

Another stochastic processes that have proven to be very useful for traffic modeling purposes, are the so-called asymptotically second-order self-similar processes. Using the same notation as before, the process $X$ is called asymptotically second-order self-similar with parameter $H = 1 - \beta/2$, $0 < \beta < \infty$, if for all $k \in \mathbb{N}_1$,

$$\lim_{m \to \infty} r^{(m)}(k) = \frac{1}{2}[(k + 1)^{2-\beta} - 2k^{2-\beta} + (k - 1)^{2-\beta}]$$  \hspace{1cm} (2.8)

Similarly as the exactly second-order self-similar processes, the asymptotically second-order self-similar processes exhibit the LRD and the slowly decaying variance properties. Namely, as shown in [TG97a], if $X$ is asymptotically second-order self-similar, then

$$\lim_{k \to \infty} \frac{r(k)}{k^{-\beta}} = c_1 \quad 0 < \beta < 1$$  \hspace{1cm} (2.9)

where $0 < c_1 < \infty$ is a constant, and

$$\lim_{m \to \infty} \frac{\text{Var} [X^{(m)}]}{m^{-\beta}} = c_2 \quad 0 < \beta < 1$$  \hspace{1cm} (2.10)

where $0 < c_2 < \infty$ is also a constant. Equivalently it can also be shown that (2.8) and (2.10) follow from (2.9) (see e.g. [TG97a]). Hence, if the process satisfies any one of the three properties, it also satisfies the other two.

Comparison of (2.5) and (2.9) shows that the main difference between the exact and asymptotically second-order self-similar process with parameter $H = 1 - \beta/2$, $0 < \beta < 1$, is that the former must have $r(k) \sim H(2H - 1)k^{-\beta}$, $k \to \infty$, whereas the latter has only $r(k) \sim ck^{-\beta}$, $k \to \infty$, where $c$ is some constant not necessarily equal to $H(2H - 1)$.

The importance of the asymptotically second-order self-similar processes for traffic modeling is further exemplified with the following two properties. If $X$ and $Y$ are two
uncorrelated asymptotically second-order self-similar processes with \( r_1(k) \sim c_1 k^{-\beta_1}, k \to \infty \) and \( r_2(k) \sim c_2 k^{-\beta_2}, k \to \infty \), respectively, then \( X + Y \) is also an asymptotically second-order self-similar process with parameter \( H = 1 - \beta/2 \) where \( \beta = \min(\beta_1, \beta_2) \). In other words, multiplexing two asymptotically second-order self-similar traffic streams, produces again an asymptotically second-order self-similar stream. Similarly, if \( X \) and \( Y \) are exactly second-order self-similar processes, then \( X + Y \) is exactly second-order self-similar with parameter \( H \), if and only if \( H_1 = H_2 = H \), otherwise \( X + Y \) is an asymptotically second-order self-similar process with parameter \( H = \max(H_1, H_2) \). Hence the self-similar traffic models used in the rest of the thesis, are based on asymptotically second-order self-similar processes.

An important consequence of the last two properties is that aggregation of self-similar traffic streams yields a traffic stream with a higher level of burstiness, i.e. higher \( H \) values. The SRD processes, on the other hand, become smoother when aggregated.

**Heavy-tailed property**

Another property which is widely considered to be closely associated with the self-similar processes is the *heavy-tailed property*. A random variable \( U \) is said to have a heavy tail if its complementary distribution function has the following form [LTWW94]:

\[
\Pr\{U \geq u\} \sim u^{-\alpha} h(u) \quad \text{as } u \to \infty
\]  

(2.11)

for \( \alpha > 0 \) and \( h \) is slowly decaying in infinity, i.e. \( \lim_{u \to \infty} h(u \tau)/h(u) = 1 \). Note that if \( \alpha \in (0,1] \), then all moments of \( U \) are infinite. In general the \( n \)-th moment will be infinite if \( n \geq \alpha \).

Heavy-tailed distributions are often used for generation of self-similar traffic patterns [LTG95, GB96, TG97b]. They have also proved to be of crucial importance to understanding the question *why is the network traffic self-similar?* Developing an approach originally proposed by Mandelbrot [Man69], Willinger et al [WTSW97] have shown that superposition of (strictly alternating or i.i.d.) ON-OFF sources, each of which exhibits the so-called Noah effect or the infinite variance syndrome [Ber94, page 211], results in a self-similar aggregate traffic. Intuitively the Noah effect means that the ON and OFF times of a single source are highly variable, covering several time scales. Mathematically speaking the Noah effect translates into heavy-tailed distributions with infinite variances, i.e. with \( 1 < \alpha < 2 \) (see (2.11)).

The findings are also supported by the analysis of two Ethernet traces at the source level, which showed that the traffic generated by the individual source destination pairs indeed exhibits the Noah effect. This is in sharp contrast to the traditional traffic modeling practice, which always assumes finite variance distributions for the ON and OFF periods, raising concern about the validity of the derived traffic engineering guidelines.

To this end another recent study [CB96] has also shown that the self-similarity of the WWW traffic results from the heavy-tailed distributions of the transmission and silent times. The authors went even further to trace the causes of such behavior. They found that the transmission times are heavy tailed primarily due to the distribution of the size of

---

5Subsection 2.3.2 shows analytically that LRD follows from the heavy-tailed distribution of the inter-arrival times of the traffic model.
the files on the Web, whereas the user think time is the major cause of the high variability of the silent periods. In other words, it seems that the roots of traffic self-similarity go back to the way people manipulate information (storage and retrieval). The heavy-tailed sizes of the Web files are similar to the Pareto\textsuperscript{10} distributions observed in social sciences, like distributions of lengths of books on library shelves or distributions of word lengths in sample texts [Man83].

This suggests that the self-similar nature of the network traffic is not a machine-induced artifact, but an inherent feature of human information processing. Therefore changes in the communication protocols and network infrastructure will probably influence the level of self-similarity, but are not likely to remove this fundamental property of the network traffic altogether.

2.3 Traffic modeling

Traffic characterization and modeling have received considerable attention from the teletraffic community because of the need to understand better the relationship between the relevant traffic parameters and the amount of network resources, which will guarantee the required level of QoS. A traffic model should contain only the most important characteristics of the actual traffic. The key is to identify those parameters which are most relevant for the network performance. The theoretical models are then fitted to the actual traffic by matching the values of the parameters in the model to the actual traffic.

There is an immense number of traffic models described in the literature\textsuperscript{11}. This section describes only the models that are used in the following chapters. The presentation is structured according to the time-scale framework described in chapter 1.

2.3.1 Connection level models

The connection level characterizes the call arrivals and their holding times. This section follows the classical approach, i.e. it proposes using the Poisson process to model the arrival process at the connection level. Poisson models are the oldest traffic models, dating back to the advent of telephony and the pioneering work of A.K. Erlang [Erl17]. A Poisson process can be characterized as a renewal process with exponentially distributed inter-arrival times:

\[ \Pr\{A_N \leq t\} = 1 - e^{-\lambda t} \]

where \( A_N \) denotes the interarrival time and \( \lambda \) denotes the call arrival rate. It can equivalently be described as a counting process, satisfying:

\[ \Pr\{N(t) = n\} = \frac{(\lambda t)^n}{n!} e^{-\lambda t} \]

where the number of arrivals in disjoint intervals \( N(t) \) are independent. The latter property is also known as the memoryless property and it greatly simplifies the analysis. It

\textsuperscript{10} The Pareto distribution is defined in section 2.3.2.
\textsuperscript{11} A good overview can be found in [FM94]
2.3 Traffic modeling

can also be shown that superposition of independent Poisson processes results again in a Poisson process with a rate equal to the sum of the rates of the independent processes [Gar94].

Poisson traffic modeling has been used very successfully for almost a century in dimensioning of POTS. The approach is based on Palm’s theorem, which basically says that under certain conditions, the aggregation of a large number of generally distributed independent traffic streams approaches a Poisson process, where the number of streams grows while the individual rates decrease in order to keep the mean constant. This theoretical result was well supported by the telephone traffic measurements of the time.

Note, however, that ATM traffic integrates telephone with data, video and multimedia traffic. The common belief in the beginning was that it should not make any difference, not at the connection level at least. Recent analysis of data and telephone traffic measurements have produced some interesting results, however. Although the user-initiated call arrivals are well-modeled with Poisson processes with fixed hourly rates [PF94], the assumption of exponentially distributed call holding times begins to break down [DMRW94]. In particular it seems that telephone call holding times are showing heavy-tailed properties, which is a characteristic for fractal processes, as discussed in section 2.2.1.

The cause of such behavior can be found in the introduction of modems and fax machines and particularly in the enormous popularity of the Internet combined with a wide deployment of PCs in households, all of which contribute to the occasionally very long call holding times. Consequently, the classical approach based on M/M/k/k system may not be as accurate as it used to be.

2.3.2 Burst level models

At the intermediate level of the BURST time scale the incoming traffic is described with its instantaneous rate. An event at this level represents a change in the rate at which the user generates cells. Particularly, the analysis at this scale is concerned with the time intervals during which the incoming cell rate exceeds the capacity of the link.

This subsection focuses on the BURST level traffic modeling, which has received a lot of attention in the last few years because of the discovery of the self-similarity phenomenon in the BURST scale traffic measurements. The presence of LRD in virtually all recent traces has raised concerns about the validity of the traffic engineering guidelines, which are based almost exclusively on SRD processes. These issues are investigated in Chapter 3. This subsection describes the traffic models that are used in the analysis. First it describes BURST scale traffic measurements. Two traffic types are considered: a WWW traffic measured on our university network and a 10-minute video sample from the movie Star Wars. Then two types of traffic models are described, traditional and self-similar, used to model the behavior of the traces. Finally the validity of the models is verified against the actual traffic measurements.

Traffic measurements

Practical measurements of the user cell rate is not a trivial matter. This is because to measure at BURST scale one has to choose a unit interval over which to average the instantaneous rate. It is not always clear how to make this choice. Sometimes there exists
Chapter 2. Characterization of broadband traffic

Figure 2.4: Measured traffic traces: a)–c) WWW traffic from the campus network of Delft University of Technology, d) VBR video traffic from Bellcore.

The measured traffic traces show an inherent mechanism from which the size of the averaging window follows naturally\(^\text{12}\), and sometimes it is simply determined by the limitations of the measuring equipment.

In this section I present two measured data sets and their statistics that are used in the performance analysis in Chapter 3. The first data set was measured on the Ethernet backbone of the Faculty of Electrical Engineering at the Delft University of Technology (TU-Delft). This was a 10 Mbit/s link that connects floors 10 to 21 of the Faculty building to the rest of the world. Between the 4th of December 1996 and the 21st of January 1997, continuous measurements were performed to determine the amount of WWW traffic. All local (internal TU-Delft) packets were discarded in order to study only the external traffic. The number of WWW packets and the packet size were logged during 120 minutes each day, in 10 ms intervals.

This resulted in 49 traces of which we selected the three busiest days. Figure 2.4 a)–c) depicts the first 10 minutes of the traces. The time series show the bursty nature of the WWW traffic. There are a lot of intervals without any WWW activity. In fact, in only about 40% of all time slots, the WWW traffic is actually present.

Table 2.2 summarizes the statistics of the traces. The Hurst parameter H was found using the variance-time plot method described in section 2.2.1. The plots are displayed in figure 2.5 a)–c). The values of 0.76 – 0.86 strongly indicate long-range dependence in

---

\(^{12}\)Like the frame size duration by video traffic.
2.3 Traffic modeling

<table>
<thead>
<tr>
<th>Parameter</th>
<th>WWW traffic traces</th>
<th>Video trace</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Date</td>
<td>Daytime</td>
</tr>
<tr>
<td></td>
<td>20 Jan 1997</td>
<td>16:00–16:10</td>
</tr>
<tr>
<td></td>
<td>7 Jan 1997</td>
<td>14:00–14:10</td>
</tr>
<tr>
<td></td>
<td>5 Dec 1996</td>
<td>10:00–10:10</td>
</tr>
<tr>
<td>Time unit, $\Delta t$ (ms)</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Peak rate (bytes/$\Delta t$)</td>
<td>12440</td>
<td>9168</td>
</tr>
<tr>
<td>Mean rate, $\mu$ (bytes/$\Delta t$)</td>
<td>288.73</td>
<td>246.69</td>
</tr>
<tr>
<td>Stand. dev., $\sigma$ (bytes/$\Delta t$)</td>
<td>814.06</td>
<td>530.89</td>
</tr>
<tr>
<td>Peak/Mean</td>
<td>57.29</td>
<td>37.16</td>
</tr>
<tr>
<td>Coef. of var., $\sigma/\mu$</td>
<td>2.8</td>
<td>2.2</td>
</tr>
<tr>
<td>Hurst parameter, $H$</td>
<td>0.86</td>
<td>0.81</td>
</tr>
</tbody>
</table>

Table 2.2: Different statistics of the measured traffic

WWW traffic, and correspond with the findings described in [CB96].

The second data set is a VBR video trace as described and analyzed in [GW94]. The data was downloaded from the Bellcore site at thumper.bellcore.com/vbr.video.trace/. This trace was obtained by researchers at Bellcore by applying an intra-frame compression similar to JPEG to the movie Star Wars. The movie runs at a speed of 24 frames per second, but since the first data set uses 10 ms units, the length of each frame is set to 40 ms\textsuperscript{13}. The time series of the first ten minutes of the movie are shown in Figure 2.4 d) and its statistics in Table 2.2.

As expected, the pictures demonstrate the fundamental differences between data and video traffic. Another difference becomes clear after generating the rate histogram by dividing the maximum bit rate of each trace over 50 equally sized bins. The bins are plotted against their probability of occurrence in figure 2.6. The WWW traces all have a high probability for the first bin (no activity), after which the probabilities decay exponentially fast to the maximum speed of the measured link (12500 bytes/10ms). For the video trace all bins lie approximately in the same range of probability except the last 12 bins, which have significant lower probabilities of occurrence, and the first 8 bins, which are empty. The differences in figure 2.6 can be explained by the fact that the VBR video trace is a realization of a rate-oriented application, which generates a continuous flow of information, whereas the WWW traces are more of an ON-OFF type on the 10 ms level\textsuperscript{14}.

This behavior also illustrates the need for application-specific traffic modeling at the burst scale. In other words, the burst level behavior of different applications is such that development of a generic traffic model encompassing all application types is cumbersome. However, we do not want to end up in a situation where we have a different model for each application. In Section 2.1 the broadband applications were divided in four traffic classes, according to their QoS requirements. For reasons already discussed, we are primarily interested in the performance of two of the four classes: LD and LL. Therefore the rest of this section focuses on different LD and LL traffic models.

\textsuperscript{13} Actually we played the movie 4% faster than intended.

\textsuperscript{14} One should also note that the WWW traces are measurements of aggregate traffic, encompassing many source-destination pairs, whereas the VBR video trace is just one instance of a single video coder.
Figure 2.5: Variance-time plots of the measured traces. a)–c) correspond to the WWW traffic measurements, and d) to the video trace.

Classical traffic models

This subsection considers traffic models that do not account for the self-similarity of the network traffic. In other words, they do not specifically model the correlation over very long lags $k$.

The proposed LD model is based on a VBR video coder model developed by Maglaris et al [MAS*88, Onv94]. The video traffic has a specific structure due to the way the coders operate. The rate process is piecewise constant, i.e. during each frame interval, the bits of one frame are packed into cells and transmitted at a constant average rate. The time-dependent behavior of the VBR coders is modeled with a \textit{first-order autoregressive} process (AR).

Let $\omega(n)$ denote the bit rate in the $n^{th}$ frame of a single source. Then according to the AR model, $\omega(n)$ depends on the bit rate in the preceding frame:

$$\omega(n) = \alpha \omega(n - 1) + \beta \xi(n)$$ \hspace{1cm} (2.12)

where $n = 1, 2, \ldots$ and where $\xi(n)$ is a normalized Gaussian random variable with mean $E[\xi(n)]=\gamma$. The constants $\alpha$, $\beta$ and $\gamma$ are found by matching the steady-state mean, variance and autocovariance function of the model, to the measured data. The obtained values are: $\alpha \approx 0.99$, $\beta \approx 0.19 \cdot 10^6$ and $\gamma \approx 0.19$. 
2.3 Traffic modeling

Figure 2.6: Normalized histograms of the measured traffic traces. a)–c) correspond to the WWW traffic measurements, and d) to the video trace.

The model is quite accurate for modeling uniform activity-level video, like videoconferencing, as found from the actual measurements [MAS+88]. It is unsuitable for analytical analysis, however. Since our analysis method known as *simpulation*\(^{15}\) combines simulation at the burst level with cell level numerical analysis, this limitation does not pose a problem.

Very popular burst level traffic models are the ON-OFF processes and their superposition [AMS82, GAN91, Rob92, RMV96]. An ON-OFF process is defined as a renewal process that alternates between periods during which it generates cells with a constant rate (ON periods) and periods with no activity or zero rate (OFF periods). This is often a good abstraction of source-level data traffic, which is a typical example of the LL traffic class. Note however that by a proper choice of the parameters, the ON-OFF processes can also be used for modeling of video traffic [MAS+88, Onv94].

The usual assumption is that ON and OFF times are *independent*, but not necessarily identically distributed. Thus the model is a simple case of a semi-Markovian process. Let the transmission rate in ON state be \(R_{\text{peak}}\). Define \(I(\text{ON})\) as the indicator function, i.e.

\(^{15}\)The *simpulation* method is described in section 3.2.1.
Chapter 2. Characterization of broadband traffic

\[ I(ON) = 1, \text{ if the source is in ON state. Then, the rate process is defined as:} \]

\[ \lambda(t) = I(ON) R_{\text{peak}} \]  \hspace{1cm} (2.13)

Further assume that the OFF and the ON periods are exponentially distributed with parameters \( \theta \) and \( \eta \), respectively. In that case \( \lambda(t) \) is a two-state Markov process characterized by its peak rate in ON state, \( R_{\text{peak}} \), and the mean OFF and ON times, \( E[t_{\text{off}}] = \theta^{-1} \) and \( E[t_{\text{on}}] = \eta^{-1} \), respectively.

The aggregated traffic is modeled as a superposition of \( N \) homogeneous ON-OFF sources which results in continuous-time Markov chain with \( N + 1 \) states. The aggregate rate process is dependent on the state \( j \) at time \( t \), and it is defined as:

\[ \Lambda(t) = j R_{\text{peak}} \]  \hspace{1cm} (2.14)

Because the superimposed processes are independent, it easy to show that the average OFF time of the aggregated process is \( E[T_{\text{off}}] = E[t_{\text{off}}] / N \). It also follows that the number of active sources at time \( t \) is binomially distributed, independently of the ON and OFF distributions, with parameters \( N \) and \( p_{\text{on}} \), where \( p_{\text{on}} \) denotes the probability that a single ON-OFF source is active, \( p_{\text{on}} = \theta / (\theta + \eta) \). Thus the average aggregate bandwidth of \( N \) independent homogeneous ON-OFF sources can be found with:

\[ E[\Lambda(t)] = p_{\text{on}} N R_{\text{peak}} \]  \hspace{1cm} (2.15)

When the number of sources \( N \) is known, one can calculate all model parameters from the statistics in Table 2.2. \( N \) was determined by using a heuristic insight, based on the histogram of the traffic trace [Har97]. The parameters of the ON-OFF sources are given in Table 2.3.

<table>
<thead>
<tr>
<th>Date</th>
<th>20 Jan 1997</th>
<th>5 Dec 1997</th>
<th>7 Jan 1997</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daytime</td>
<td>16:00 – 16:10</td>
<td>10:00 – 10:10</td>
<td>14:00 – 14:10</td>
</tr>
<tr>
<td>Peak rate in ON state ( R_{\text{peak}} ) [kbytes/s]</td>
<td>60</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>Number of sources ( N )</td>
<td>28</td>
<td>26</td>
<td>16</td>
</tr>
<tr>
<td>Mean ON period ( E[t_{\text{on}}] ) [ms]</td>
<td>1158.92</td>
<td>857.74</td>
<td>516.96</td>
</tr>
<tr>
<td>Mean OFF period ( E[t_{\text{off}}] ) [ms]</td>
<td>20.27</td>
<td>20.98</td>
<td>13.63</td>
</tr>
<tr>
<td>Average rate per source [kbytes/s]</td>
<td>1.031</td>
<td>1.433</td>
<td>1.542</td>
</tr>
</tbody>
</table>

Table 2.3: Parameters of the ON-OFF traffic models

The two traffic models described above are complementary. The AR model has continuous state space and discrete time, whereas the ON-OFF model is a continuous-time process with discrete state space. It would be very difficult to develop a traffic model that integrates both of these models into a single stochastic process. That is not necessary, however, since one of the advantages of the simulation method is that it can easily incorporate traffic models with different structure into the analysis.
2.3 Traffic modeling

Self-Similar traffic models

One of the objectives of this thesis is to investigate the impact of the traffic self-similarity on the performance of the ATM network. In order to achieve this goal, traffic models able to capture this property are necessary. A natural first step is to determine the time scale of interest.

The resolution of virtually all self-similar traces is 10 ms, which is well above the typical cell level time scale of a few $\mu$s. Furthermore, according to [CB96, LTWW94, LTWW93], it is reasonable to expect that at the extreme aggregation levels, high as well as low, the self-similar properties of the traffic will begin to disappear. On the other hand, the BURST level variations are probably determinative for the network performance. Consequently the BURST level seems the most appropriate time scale for modeling the self-similarity phenomenon.

From a modeling viewpoint, self-similarity does not imply complex and highly parameterized traffic models. On the contrary, as suggested in [LTWW94], the self-similarity encountered in the network traffic measurements can be captured with parsimonious traffic models like fractional Brownian motion (fBm) [PSS96, LEWW95, LAEW95, Nor94], fractional ARIMA processes (fARIMA) [GW94, RH96] or fractal processes [RL96, RE96]. This subsection describes two such models, one based on the fARIMA process, used for modeling LD traffic, and the other based on a renewal process with heavy-tailed interarrival times, used for modeling LL traffic.

The employed LD traffic model is a slightly adapted version of the model developed by Garrett and Willinger in [GW94]. The model is based on a fARIMA$(0,d,0)$ process which is shown to be asymptotically self-similar in [Ber94], for $0 < d < \frac{1}{2}$. Hosking provides an algorithm for generating fARIMA$(0,d,0)$ sequences (for details see [GW94, Hos84]). The algorithm produces an LRD sequence $\{X_n\}$ of Gaussian marginals, with zero mean and unit variance. In order to match the model to the measured traffic, the marginal distribution of $\{X_n\}$ is transformed according to the following mapping:

$$Y_k = F_{\gamma/P}^{-1}(F_N(X_k)),$$

where $F_N$ is the cumulative distribution function (CDF) of a Gaussian distribution and $F_{\gamma/P}^{-1}$ is the inverse CDF of a combined Gamma/Pareto (G/P) distribution\(^{16}\) [GW94].

The G/P distribution is suggested as the best fit for the rate distribution of the measured traffic. Normally one would continue by using the G/P distribution with the appropriate parameters as a traffic model. However, since there is an empirical trace available, we use the histogram of the empirical trace instead of the G/P distribution. Thus, for each $X_k$ produced by the algorithm, one should first calculate its cumulative probability from the Gaussian CDF, and then select the appropriate traffic rate $Y_k$, corresponding to the bin with the next higher cumulative probability.

The second self-similar model is based on a model developed by Grossglauser and Bolot in [GB96]. The traffic model is described by a random process $\{Z_t\}$, representing the cell

\(^{16}\)The combined Gamma/Pareto distribution is actually a Gamma distribution with an Pareto tail.
rate at time \( t \). It is assumed that the rate process is piecewise constant and it takes on
values from a discrete set \( \{\lambda_1, \lambda_2, \ldots, \lambda_M\} \). The rate changes according to the arrivals
of a stationary point process \( \{\tau_n\} \), i.e. the rate remains constant during the interarrival
intervals of \( \{\tau_n\} \). Hence:

\[
Z_t = \lambda(n) \quad \text{for} \quad \tau_n \leq t \leq \tau_{n+1}.
\]

(2.17)

The interarrival times \( T_n = \tau_{n+1} - \tau_n \) are i.i.d. with a probability distribution function
\( F_T(t) = \Pr\{t \leq T_n\} \). It is further assumed that the rate process is i.i.d. as well, with a
probability mass function given by \( \Pr\{\lambda(n) = \lambda_i\} = \pi_i \). Note that by assuming \( \lambda(n) \in \{0, R_{\text{peak}}\} \) and \( F_T(t) = 1 - e^{-\lambda t} \), the model becomes the ON-OFF source model discussed
in the previous subsection.

In order to show self-similarity, one has to prove that the model satisfies any of the
properties discussed in section 2.2.1. Therefore I first derive the autocovariance function
of \( Z_t \). The autocovariance function of any stochastic process is defined by [Gar94]:

\[
\rho(t) = \mathbb{E}[Z_0 Z_t] - (\mathbb{E}[Z_0])^2
\]

(2.18)

where the first term on the left-hand side can be written as:

\[
\mathbb{E}[Z_0 Z_t] = \sum_{i=1}^{M} \sum_{j=1}^{M} \lambda_i \lambda_j \Pr\{Z_0 = \lambda_i, Z_t = \lambda_j\}
\]

\[
= \sum_{i=1}^{M} \lambda_i^2 \Pr\{Z_0 = \lambda_i, Z_t = \lambda_i\}
\]

\[
+ 2 \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} \lambda_i \lambda_j \Pr\{Z_0 = \lambda_i, Z_t = \lambda_j\}
\]

(2.19)

The first term on the right-hand side of the equation (2.19) corresponds to the event
when the rates at time 0 and \( t \) are identical. To calculate this term two cases have to be
considered: when 0 and \( t \) lie in the same interarrival interval of the point process, and
when they lie in different intervals. Let \( p_0(t) \) denote the probability that 0 and \( t \) lie in
the same interval, or equivalently, that there is no arrival in \([0, t]\), \( p_0(t) = \Pr\{T_n \geq t\} \). Then,
taking into account that \( \lambda_i \)'s in different intervals are independent, it follows that:

\[
\sum_{i=1}^{M} \lambda_i^2 \Pr\{Z_0 = \lambda_i, Z_t = \lambda_i\} = p_0(t) \sum_{i=1}^{M} \pi_i \lambda_i^2 + (1 - p_0(t)) \sum_{i=1}^{M} \pi_i^2 \lambda_i^2
\]

(2.20)

The second term in (2.19) corresponds to the case when the rates at time 0 and \( t \) are
different. This means that there was an arrival in this interval, which implies that \( Z_0 \) and
\( Z_t \) are independent. Hence, \( \Pr\{Z_0 = \lambda_i, Z_t = \lambda_j\} = \pi_i \pi_j \).

The second term in (2.18) is actually the square mean of \( Z_t \), which can also be written as

\[
(\mathbb{E}[Z_0])^2 = (\sum_{i=1}^{M} \pi_i \lambda_i)^2 = \sum_{i=1}^{M} \pi_i^2 \lambda_i^2 + 2 \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} \pi_i \pi_j \lambda_i \lambda_j.
\]

(2.21)
By combining (2.19), (2.20) and (2.21) with (2.18), we arrive at:

\[ \varphi(t) = p_0(t) \sum_{i=1}^{M} \pi_i \lambda_i^2 + (1 - p_0(t)) \sum_{i=1}^{M} \pi_i^2 \lambda_i^2 - \sum_{i=1}^{M} \pi_i^2 \lambda_i^2 \]

\[ = p_0(t) \sum_{i=1}^{M} \lambda_i^2 \pi_i (1 - \pi_i). \quad (2.22) \]

In order to complete the derivation we need to specify the \( p_0(t) \). Applying some renewal theory results [Kle75], and noting that \( p_0(t) \) is equivalent to the probability that the residual life of the interarrival time \( T_n, \tau_{\text{res}} \), exceeds \( t \), it follows that:

\[ p_0(t) = \Pr\{\tau_{\text{res}} \geq t\} = \int_t^{\infty} \frac{1 - \mathcal{F}_T(t)}{E[T_n]} \, dt \quad (2.23) \]

We now specify the interarrival time distribution \( \mathcal{F}_T(t) \), to follow a translated\(^{17}\) Pareto distribution, defined by [GB96]:

\[ \mathcal{F}_T(t) = 1 - \left( \frac{\theta}{t + \theta} \right)^\alpha, \quad (2.24) \]

where \( 1 < \alpha < 2 \) and \( \theta > 0 \). The Pareto distribution is the simplest heavy-tailed distribution and for \( \alpha < 2 \) it has infinite variance. The mean can be found by integrating the complementary distribution function from 0 to \( \infty \), yielding \( E[T_n] = \frac{\theta}{\alpha - 1} \). Then the autocovariance function, \( \varphi(t) \), can be found by solving (2.23), which yields

\[ p_0(t) = \Pr\{\tau_{\text{res}} \geq t\} = \left( \frac{t + \theta}{\theta} \right)^{-\alpha + 1}. \]

By substituting the latter result in (2.22), we obtain

\[ \varphi(t) = \left( \frac{t + \theta}{\theta} \right)^{-\alpha + 1} \sum_{i=1}^{M} \lambda_i^2 \pi_i (1 - \pi_i). \quad (2.25) \]

Thus \( \varphi(t) \) behaves as \( t^{-\alpha + 1} \). Since the correlation coefficient is defined as \( r(t) = \frac{\varphi(t)}{\sigma^2} \), it follows that: \( r(t) \sim ct^{-\alpha + 1} \), where \( c \) is a constant independent of \( t \). In other words, according to (2.9), \( Z_t \) is asymptotically long-range dependent, or equivalently asymptotically second-order self-similar, with Hurst parameter \( H \) defined by \(-\alpha + 1 = -(2 - 2H)\), i.e. \( H = \frac{3-\alpha}{2} \).

The parameters of the model are specified by matching the rate distribution \( \Pi = \{\pi_1, \pi_2, \ldots, \pi_M\} \), with the histogram of one of the traffic traces in Figure 2.4. The \( \theta \) is computed by equating the mean interval duration \( T_n \) and the average number of consecutive samples in the trace that fall within the same histogram bin. The Hurst parameter is taken from the appropriate column in Table 2.2.

In summary, the model allows control of the first-order statistics through the marginal rate distribution \( \Pi \) and the level of self-similarity through the Hurst parameter \( H \). Henceforth this model will be called the Pareto rate model.

\(^{17}\)The standard Pareto distribution is defined by \( \mathcal{F}_T(x) = 1 - \left( \frac{x}{\theta} \right)^\alpha \) for values of \( x \geq \theta \). Since in this case \( t \in [0, \infty) \), the random variable is translated \( \theta \) units back.
2.3.3 Verification of the burst level models

This subsection compares the synthetic traffic generated by both traditional and self-similar BURST level models with the measured traffic traces, in order to determine the quality of the proposed traffic models. We are particularly interested in the $H$ values estimated from the synthetically generated self-similar traffic.

<table>
<thead>
<tr>
<th>Type of model</th>
<th>WWW traffic</th>
<th>VBR video</th>
</tr>
</thead>
<tbody>
<tr>
<td>traditional</td>
<td>ON-OFF model</td>
<td>AR model</td>
</tr>
<tr>
<td>self-similar</td>
<td>Pareto rate model</td>
<td>fARIMA model</td>
</tr>
</tbody>
</table>

Table 2.4: The traffic models that are analyzed in this subsection.

The traffic models discussed in this subsection are given in table 2.4. The models are compared according to the type of traffic they model. The comparison is based on three different types of plots:

Traffic intensity plots allow the user to determine the quality of the synthetic traffic by quick visual inspection.

Normalized autocovariance plots allow the user to compare the short-term correlation structure of the synthetic and measured traces.

Variance-time plots allow the user to compare the long-term correlation structure of the synthetic and measured traces.

The quality of the synthetic WWW traces is discussed first. Figure 2.7 depicts both traditional and self-similar synthetic traces. By comparing these graphs with figure 2.4, one can easily see that the Pareto rate trace is very different from the measured traffic, whereas the ON-OFF trace resembles the measured traffic better, although it has smaller peaks and almost no periods of inactivity.

![Traffic intensity plots](image)

Figure 2.7: Traffic intensity plot of the synthetically generated WWW traces: left- with traditional ON-OFF model, right- with Pareto rate model.
2.3 Traffic modeling

The autocovariance plots are given in figure 2.9 (right-hand column graphs). The figure shows that the correlation in the actual traffic decreases very fast within the first few lags and then remains very low for larger lags. The Pareto rate trace, on the other hand shows strong correlations which decrease very slowly with the increase of time. The ON-OFF trace shows similar behavior as the actual traffic, except that after the first few legs the correlation effectively drops to zero.

![Traffic intensity plots](image)

Figure 2.8: Traffic intensity plot of the synthetically generated VBR traces: left- with traditional AR model, right- with fARIMA model.

The variance-time plots are given on figure 2.10 (left-hand column graphs). The estimated values of $\beta$ and $H$ are given in table 2.5. It is clear that both the empirical and the synthetic Pareto rate trace exhibit LRD, whereas the ON-OFF trace does not, which confirms our expectations.

<table>
<thead>
<tr>
<th>WWW</th>
<th>$\beta$</th>
<th>$H$</th>
<th>VBR video</th>
<th>$\beta$</th>
<th>$H$</th>
</tr>
</thead>
<tbody>
<tr>
<td>WWW trace</td>
<td>0.28</td>
<td>0.86</td>
<td>Star War trace</td>
<td>0.5</td>
<td>0.75</td>
</tr>
<tr>
<td>Pareto rate trace</td>
<td>0.34</td>
<td>0.83</td>
<td>fARIMA trace</td>
<td>0.51</td>
<td>0.75</td>
</tr>
<tr>
<td>ON-OFF trace</td>
<td>0.95</td>
<td>0.53</td>
<td>AR trace</td>
<td>0.80</td>
<td>0.60</td>
</tr>
</tbody>
</table>

Table 2.5: Hurst parameters estimation for the empirical and synthetic traces.

Similar conclusions can also be drawn from the plots of the synthetic video traces (see figure 2.8 and figure 2.10). The AR trace appears to be a reasonable representation of the measured VBR video trace, except for the lack of LRD properties (see figure 2.10). For the fARIMA trace exactly the opposite holds; only the LRD property agrees well with the empirical trace, but the strong short-term correlation, typical for the video traffic, is completely absent (see figure 2.10).

In summary, the self-similar models represent the LRD characteristics reasonably accurately, whereas the SRD characteristics are represented very poorly. The opposite holds
for the traditional models. Although both types of models have shortcomings, the traditional models seem to be more accurate\(^{18}\). Since the quality of a traffic model should not be judged only by its resemblance to the actual traffic, the next chapter analyzes the performance of all models discussed here.

### 2.3.4 Cell level models

Cell level models are used in analysis of mechanisms that influence individual ATM cells, like flow control algorithms, priority queues or switch architectures. There are two basic classes of cell level models [RMV96]:

- models based on renewal processes, like the Poisson or Bernoulli process.
- models based on Markov modulated processes, like the Discrete time Batch Markovian Arrival Process (D-BMAP).

The second class encompasses more complex models with a lot of parameters that often do not have any physical interpretation. Therefore I prefer the Bernoulli process as a cell arrival process at the cell level. A Bernoulli process is a discrete time process which takes the value 1 with a probability \( p(t) \) or 0 with probability \( 1 - p(t) \), for each time slot independently. Intuitively, since each time slot an ATM link either transmits a cell or is idle, a Bernoulli process seems the appropriate choice.

The choice is based on the assumption that the correlation in the arrival streams can be modeled by changing the cell arrival probability at the slower time scale of the burst level. At cell level, due to the jitter caused by the multiplexing of many traffic streams inside the ATM network, successive cell arrivals can be considered independent, hence the Bernoulli model.

### 2.4 Conclusion

Determination of the application performance requirements and the traffic description are necessary building blocks for ATM network analysis and design. This chapter considers both issues. The first section discusses a wide variety of applications and their QoS requirements that are likely to play a significant role in the future broadband networks. To simplify the network operation, the applications are classified into four categories, based on their loss and/or delay tolerance (see figure 2.1):

- **LX** applications, both delay- and loss-sensitive, hence they have the highest priority,
- **LD** applications, sensitive to delay (jitter), but tolerant to moderate losses,
- **LL** applications that are sensitive to cell loss, but tolerant to larger delays, and
- **HX** applications, which are tolerant to relatively high losses and delays, and consequently have the lowest priority.

\(^{18}\) This conclusion is later confirmed by the performance analysis experiments in Chapter 3.
Since there is little to gain from multiplexing of LX sources because of their stringent QoS requirements, and the higher priority cells will hardly, if ever, notice the low priority HX class, the most promising classes for achieving higher efficiency gains are the LD and LL classes.

Section 2.2 considers the recently discovered self-similar property of the broadband traffic that is not accommodated in the current traffic modeling and analysis techniques. It introduces the mathematical definition of self-similar processes and discusses some of their properties. It also argues that such traffic behavior is a consequence of the way people process information, and not of the communication protocol structure or network topology.

Section 2.3 considers various traffic models, used throughout the rest of the thesis. The models are divided into three categories, according to the time scale relevant for the performance analysis. The preferred traffic model at the CONNECTION level is the Poisson model, mainly because the user-initiated call arrivals are still very well-modeled by a Poisson process. At the BURST level, there is a need for application-specific traffic modeling. Therefore following the approach of section 2.1, the chapter presents different models for both LD and LL traffic. In addition, one can distinguish between traditional traffic models, which are short-range dependent, and self-similar models, which are long-range dependent.

Section 2.3.3 examines the quality of the proposed BURST level models by comparing the synthetically generated traces with the measured traces. The examination has shown that the traditional models provide a reasonably accurate representation of the real network traffic, except for the self-similarity property, which is represented very poorly. The self-similar traffic models show exactly the opposite behavior.

Finally, at the CELL level, the Bernoulli process is proposed as the ubiquitous CELL level traffic model.

The various traffic models described in this chapter are given in Table 2.4.

<table>
<thead>
<tr>
<th>Time scale</th>
<th>Traffic model</th>
<th>Correlation structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONNECTION</td>
<td>Poisson process</td>
<td>$r(k) = 0, \forall k &gt; 0$</td>
</tr>
<tr>
<td>BURST</td>
<td>AR process</td>
<td>$\sum_k r(k) &lt; \infty$</td>
</tr>
<tr>
<td>ON-OFF sources</td>
<td>$\sum_k r(k) &lt; \infty$</td>
<td></td>
</tr>
<tr>
<td>fARIMA process</td>
<td>$\sum_k r(k) \to \infty$</td>
<td></td>
</tr>
<tr>
<td>Pareto rate model</td>
<td>$\sum_k r(k) \to \infty$</td>
<td></td>
</tr>
<tr>
<td>CELL</td>
<td>Bernoulli process</td>
<td>$r(k) = 0, \forall k &gt; 0$</td>
</tr>
</tbody>
</table>

Table 2.6: Classification of various traffic models used in the thesis. The $r(k)$ denotes the autocorrelation function of the traffic (model).
Figure 2.9: Variance-time and covariance plots, left and right column respectively, of the measured, traditional and self-similar WWW traces (first, second and third row, respectively).
2.4 Conclusion

Figure 2.10: Variance-time and covariance plots, left and right column respectively, of the measured, traditional and self-similar VBR video traces (first, second and third row, respectively).
Chapter 2. Characterization of broadband traffic
Chapter 3

ATM resource sharing

One of the essential requirements of an ATM network is that it efficiently shares its bandwidth and memory resources between various traffic streams, while maintaining control over the QoS offered to the user. In principle ATM offers three types of QoS guarantees [Kur93, dV KW95]: deterministic, statistical and best-effort guarantees. In the deterministic case, the network guarantees the negotiated performance of all cells within a session. For instance, if the cell loss is used as performance metric, a deterministic guarantee would be that no cells are being lost. In the statistical case, the network guarantees the performance to only a fraction of the cells within a session, i.e. no more than a certain fraction of cells will experience performance below a specified value. For example, using cell loss as a performance metric, a statistical guarantee would be that only 1% of the cells are being lost. In the best-effort case, the network provides no QoS guarantees to the user. The experienced quality depends on the level of congestion in the network.

The nature of the QoS guarantees also depends on the applied traffic control mechanisms. In general we distinguish between preventive and reactive mechanisms. The former relay on an “a priori” description of the traffic and policing of the accepted connections, in order to ensure compliance with negotiated traffic contract. The later are based on the ability of the source to adapt its sending rate according to the current levels of network congestion (see description of ABR flow control in section 1.4.2). This chapter considers preventive techniques only, because the high bandwidth-delay product typical for ATM networks, renders the reactive control impractical for most real-time, high-bandwidth users.

There are several approaches for performing preventive traffic control in ATM networks, depending on the type of guarantees the control mechanisms provide. One approach is to limit the combined rate of the traffic streams on each link, to be less or equal to the link capacity [dV KW95]. This can be achieved by employing a leaky bucket policer (for description see [RMV96, dP95]) at the UNI. A leaky bucket with a constant token arrival rate, $\rho$, and a finite token buffer size, $\sigma$, will limit its output stream to bursts of size $\sigma$ and a mean rate never exceeding $\rho$. Hence depending on the $(\sigma, \rho)$ traffic description, the network can reserve bandwidth and buffer resources that satisfy deterministic end-to-end QoS requirements (no loss and bounded delay). A traffic stream that complies fully with the $(\sigma, \rho)$ description, would incur no additional loss or delays at the UNI.
However, a stream with unknown or a characterization that widely differs from the \((\sigma, \rho)\) description, may incur cell loss as well as random delay at the UNI, due to the operation of the leaky bucket. Another disadvantage of this approach is that as the sources become more bursty the resource utilization must drop in order to satisfy the deterministic QoS requirements. The effect is especially severe when the sources' peak rates are in the same order of magnitude as the link capacity. Nevertheless, the simplicity and the ability to provide deterministic guarantees, makes this approach appealing, especially for control of LX class of applications, whose stringent QoS requirements prevent any substantial multiplexing gains.

The second approach allows the combined arrival rate of the traffic streams to exceed the residual link capacity\(^1\) with a certain non-negligible probability, relying on large buffers in the network nodes, to absorb the momentary input rate excesses. It is based on approximate statistical traffic descriptors\(^2\), as a means to allocate network resources for connections which can tolerate statistical QoS requirements. That makes it attractive to both LL and LD applications. The main advantage of the approach is that it allows higher resource utilization, because by allowing occasional cell loss and random delays, more sources can share the link capacity. The downside is that the resource allocation and policing are more complicated. In particular the delay and jitter requirements of the LD applications impose use of short buffers, whereas LL applications need large buffers in order to keep the fraction of lost cells low enough. Therefore in order to efficiently support both traffic classes, some kind of priority(non-FIFO) queuing discipline is necessary.

The third approach is concerned with the control of the best-effort traffic (the HX application class). The best-effort traffic is serviced only when there is neither deterministic (LX) nor statistical (LD and LL) traffic present in the buffers. Hence it can be buffered both at the UNI and at the network nodes, without degrading the service of the other classes. High congestion levels could be avoided by using higher layer end-to-end control protocols.

Hence, giving head of line priority in the network switches and reserving bandwidth, is a simple and practical solution for the LX class of applications, providing deterministic QoS guarantees. Connections that can tolerate statistical performance guarantees are handled by some kind of priority queuing strategy that distinguishes between various QoS requirements, and by statistical allocation of resources. The HX traffic class is allocated buffer space and offered the left-over (idle) bandwidth.

This chapter focuses on control mechanisms for traffic tolerating statistical QoS guarantees. In particular, we analyze the performance of a priority queuing mechanism known as the LDOLL queue, which multiplexes the LD and LL traffic in such a way that improves the performance of both classes. Section 3.1 is concerned with the CELL level performance of the LDOLL queue. The first subsection presents the mathematical model of the LDOLL queue with a single outgoing link and summarizes previous research results, whereas the second subsection generalizes the LDOLL principles to an ATM switch with multiple outlets, sharing the same output buffer. Section 3.2 analyses the BURST level performance

\(^{1}\)The capacity available after subtracting the reserved bandwidth for the deterministic traffic from the link rate.

\(^{2}\)See the definition of the effective bandwidth in section 4.1.
of the LDOLL queue under more realistic conditions (real traffic and large buffers). It also investigates the impact of the self-similar property on the performance of both FIFO and LDOLL buffers.

The results in section 3.1 are partly given in [AS91, Awa94]. The generalization of the LDOLL discipline is an original contribution, as well as the results in section 3.2 (are also new).

3.1 Cell level analysis

As already said, in order to satisfy the statistical QoS requirements of both LD and LL traffic streams and still realize high levels of resource utilization, some form of priority queuing strategy should be implemented at the cell level. A priority queuing strategy is basically a rule for sharing the limited buffer- and transmission-capacity. Optimal sharing asks for best storage and retrieval policies: the storage policy specifies the priorities in using the limited buffer capacity, the retrieval policy specifies the priorities in using the limited transmission capacity.

There are different strategies proposed in the literature (see [JS96, RMV96] for extensive lists of references). This chapter analyses the so-called Low Delay Or Low Loss (LDOLL) queuing strategy, first proposed by Awater and Schoute in [AS91], which distinguishes between two traffic classes with complementary QoS requirements: LD and LL class. The LDOLL queue exploits the complementarity of the QoS requirements of the two classes, to achieve lower cell loss probability for the LL cells and in the same time lower delay and delay variance for the LD cells. The optimal performance is achieved by assigning storage priority to the LL class and conditional retrieval priority to the LD class.

A comparison between the performance of the single outlet LDOLL queue with a large variety of other priority strategies, was done in [JS96]. It was shown that the LDOLL queue outperforms all work conserving\(^3\) policies, both in stationary and time-varying regimes.

This section is concerned with the cell level performance analysis and optimization of the LDOLL queuing strategy. It assumes that the burst level arrival rate is constant and the queuing system is in equilibrium. The first subsection describes the architecture of a general ATM switching element. Subsection 3.1.2 presents the mathematical model of the single outlet LDOLL queue, together with some performance results based on the equilibrium analysis. Subsection 3.1.3 explores the possibilities of expanding the concepts of the single outlet LDOLL queue to the multiple outlet case, where the outlets are sharing a common buffer. One is primarily interested in the optimal storage- and retrieval-policies and the potential performance improvements.

3.1.1 ATM switching element architecture

The model of the ATM switching element presented here can be viewed as a basic building block of an ATM network. The schematic of the switching element is given on the picture below (see fig 3.1).

\(^3\)A definition of a work conserving policy is given on page 54.
Figure 3.1: A generic schematic diagram of an ATM switching element.

The switch contains $N$ input links and $M$ output links. An cell arrived at any of $N$ input links can be switched to any of the $M$ output links. The duration the time slot corresponds to the time needed to put an ATM cell on the transmission link, and it is determined by the speed of the link. For example, on a 150 Mbit/s ATM link, the length of the time slot is only about 2.8 $\mu$s. The model also assumes that all links are cell synchronized and operate at the same speed. Thus the system can be described at a discrete time-scale.

The functional blocks $CR_1 \ldots CR_N$ represent the cell receivers. Their function is to deserialize and store every incoming ATM cell into the input buffers, $IB_1 \ldots IB_N$. The deserialization itself, provides sufficient time to process the cell header\footnote{ATM cells with corrupted headers and unassigned cells are discarded.}, and to distinguish between cells of different classes. The cell enqueuer $CE$, takes the cells from the input buffers and stores them into the common buffer $CB$, following certain storage policy. The cell server, $CS$, retrieves cells from the $CB$ according to some predefined retrieval policy and puts them into the output buffers, $OB_1 \ldots OB_M$. Each time slot the cells are forwarded to the cell transmitters $CT_1 \ldots CT_M$, where every cell is transformed back to the serial form and then transmitted onto the outgoing link. As a consequence, the switch introduces an additional delay of one time slot, known as the transmission delay. It is an inherent property of the cell switching and it is not considered in the analysis.

Where is the actual switching done, is a question which deserves more attention. In the model presented here, it can be done either by the $CS$ or by the $CE$. Depending which of the two actually performs the operation, one can distinguish between two different types of buffering: input buffering, output buffering. In the input buffering case, the $CS$ does the switching and every input link has its own partition of the $CB$. When the switching is done by the $CE$, then depending on how the storage in the common buffer is organized, one can distinguish between: separate output buffering, where a partition of the $CB$ is designated to each output link, and shared output buffering, where the buffer
resources are shared between all output links.

The analysis of all three methods (see [HK88]) shows that the input buffering, although rather simple to implement, performs inferior compared to the other two methods. The separate and shared output buffering both realize the optimal throughput/delay performance, and while the shared output buffering increases the average cell delay, the combined performance (loss probability + average delay) is favorable with regard to the separate output buffering. The tradeoff is the increased implementation complexity of the shared output buffering.

The rest of the chapter focuses solely on the output buffering method. The next subsection considers a special case of an LDOLL queue with $N$ arbitrary and $M = 1$, i.e. the separate output buffering method. The general case where both $N$ and $M$ are arbitrary is considered subsection 3.1.3. It is assumed that both the CE and the CS can distinguish between the LD and the LL cells.

### 3.1.2 The single outlet LDOLL queue

This section describes the modeling and the operation of the LDOLL queue with a single outgoing link. Since there are two types of cells in the buffer, the single outlet LDOLL queue is modeled as a two-dimensional Markov chain. The state of the queue is defined with the random vector variable $N(t) = (N_{LD}(t), N_{LL}(t))$, where $N_{LD}(t)$ and $N_{LL}(t)$ denote the number of LD or LL cells, respectively. The state space, $\Xi$, is defined as:

$$\Xi = \{n \in \mathbb{N}^2 | n^\top \cdot 1 \leq Q\},$$  \hspace{1cm} (3.1)

and its size is determined by the buffer size, $Q$, as: $|\Xi| = (Q/2+1)(Q+1)$. Equation (3.1) simply states that the total number of cells in the queue (LD and LL), can never exceed $Q$. Consequently the state space diagram has a triangular form (see figure 3.2).

In order to fully specify the LDOLL queue, one has to define the arrival process, the storage policy and the retrieval policy. In the following, each subject is discussed in some more detail.

#### The arrival process

Every time slot $t$, $t \in \mathbb{N}$, a new cell arrives at each CR. After processing only the non empty cells are transferred to the IBs. Let the vector random variable $A(t) = (A_{LD}(t), A_{LL}(t))$ denote the number of LD and LL cells, respectively, in all IBs at time slot $t$. The set of all possible arrival vectors is denoted by $\Phi$ and it is defined with:

$$\Phi = \{a \in \mathbb{N}^2 | a^\top \cdot 1 \leq N\}$$  \hspace{1cm} (3.2)

Using the same argument as in subsection 2.3.4, the arrival process is modeled as a Bernoulli process, with two parameters, $p_k(t)$ and $r_k(t)$. The $p_k(t)$ is called the link load and it denotes the probability of a cell arrival at IB$_k$, during time slot $t$, and $r_k(t)$ is called the mixing ratio and it denotes the probability that the arriving cell is of LD type. Using this model and assuming independence between the input cell streams, the joint probability distribution of $A_{LD}(t)$ and $A_{LL}(t)$ is readily found with some book keeping.
Let $\zeta_k((i,j), t)$ be the probability that $i$ LD and $j$ LL cells are present in the IB$_k$ at time slot $t$. From the definition it follows that $Pr\{A(t) = a\} = \zeta_1(a, t)$, where $\zeta_1(a, t)$ can be found with the following recursive procedure:

$$
\zeta_k((i,j), t) = r_k(t) \cdot p_k(t) \cdot \zeta_{k+1}((i-1,j), t) + (1-r_k(t)) \cdot p_k(t) \cdot \zeta_{k+1}((i,j-1), t) + (1-p_k(t)) \cdot \zeta_{k+1}((i,j), t)
$$

with initial conditions:

$$
\zeta_{N+1}((i,j), t) = \begin{cases} 
1 & \text{if } i = j = 0 \\
0 & \text{otherwise}
\end{cases} \quad (3.4)
$$

$$
\zeta_k((i,j), t) = 0 \quad \forall i < 0, \forall j < 0, \quad k \in \{1, 2, \ldots, N\}. \quad (3.5)
$$

In order to calculate the performance measures of interest, we also need the mean LD and LL load. They can be found by:

$$
\lambda_{LD}(t) = E[A_{LD}(t)] = \sum_{k=1}^{N} r_k(t) \cdot p_k(t) \quad (3.6)
$$

$$
\lambda_{LL}(t) = E[A_{LL}(t)] = \sum_{k=1}^{N} (1-r_k(t)) \cdot p_k(t). \quad (3.7)
$$

The calculation is based on the mutual independence of the input streams. The mean load is first calculated for one input link and then summed over all IBs, for each class. Similarly the variance of the number of LX cells arrived in time slot $t$, can be found as:

$$
\text{Var}[A_{LD}(t)] = \sum_{k=1}^{N} r_k(t) \cdot p_k(t) \cdot [1-r_k(t)p_k(t)] \quad (3.8)
$$

$$
\text{Var}[A_{LL}(t)] = \sum_{k=1}^{N} r_k(t) \cdot p_k(t) \cdot [1-(1-r_k(t)p_k(t)]. \quad (3.9)
$$

The storage policy

The single outlet LDOLL queue mixes two types of cells in its common buffer. The cells are organized in two linked lists, each keeping track of the order in which the cells of the respective class have arrived. The storage policy determines which cells are stored in the CB and which are blocked, when there is insufficient buffer space$^6$.

The basic principle of the LDOLL queue is that it should improve the performance of both LD and LL traffic classes. Since the LL class requires small cell loss probability and the LD class small cell delay and delay variation, it is obvious that the optimal rule is to give the LL cells both storage and push-out (replacement) priority over the LD cells. The

$^6$Note that the last definition implicitly implies that the storage and the retrieval policies of the LDOLL queue, are work conserving, i.e. the CE never refuses an arriving cell (work) when there is still free place in the CB and the CS always retrieves a cell from the common buffer, provided that there is one.
3.1 Cell level analysis

push-out priority means that when a newly arrived LL cell sees a full buffer, it is allowed to replace (push-out) some of the LD cells that are already in the buffer. This minimizes the LL loss probability while increasing the LD cell loss. However, since the LD cells can tolerate higher losses this is not a concern.

In order to write this in a mathematical form, I introduce an additional random vector variable. Let \( S(t) = (S_{LD}(t), S_{LL}(t)) \) denote the number of LD and LL cells respectively, that are retrieved from the CB at the beginning of each time slot. Then the evolution of the Markov chain, which models the behavior of the LDOPLL queue, is described with the following equations:

\[
\begin{align*}
N_{LL}(t+1) &= \min(Q, N_{LL}(t) - S_{LL}(t) + A_{LL}(t)) \\
N_{LD}(t+1) &= \min(Q - N_{LL}(t+1), N_{LD}(t) - S_{LD}(t) + A_{LD}(t)),
\end{align*}
\]  

(3.10)

or by using vector notation for simplicity:

\[
N(t + 1) = \nu(N(t), S(t), A(t)).
\]

(3.11)

Note that the state evolution equations specify only the relation between the numbers of cells present, retrieved and arrived at the buffer. In particular, they do not specify which LD cells should be replaced. A broader discussion about this issue can be found in [Awa94]. For the purpose of the analysis here, it is assumed that the LL cells are stored first, which is the worst case scenario with regard to the delay (variance) of the LD cells.

The retrieval policy

A retrieval policy is a rule which prescribes which cells should be retrieved by the CS, at the beginning of each time slot \( t \). In the single outlet case \( (M = 1) \), the CS has three options: to retrieve an LD cell \( (s = (1, 0)) \), to retrieve an LL cell \( (s = (0, 1)) \) or to retrieve no cell \( (s = (0, 0)) \). Assuming that the retrieval policy is work conserving, in states where the queue is empty or it contains only one type of cells, the decision is straightforward. However in all other states it is not clear what is the best decision with regard to the performance of both classes.

Awater and Schoute [AS91] have proven that the policy which minimizes both the LL loss probability and the mean LD cell delay in an LDOMLL queue with a single outlet, belongs to a set of \( Q \) so-called LDOMLL threshold policies. An LDOMLL threshold policy, \( \mathcal{R}(\Delta) \), is defined as a deterministic\(^6\) policy which gives retrieval priority to the LD cells as long as the number of LL cells is below the threshold \( \Delta (n_{LL} < \Delta) \), and retrieval priority to the LL cells when the number of LL cells exceeds \( \Delta (n_{LL} \geq \Delta) \). Figure 3.2 gives an example of an LDOMLL threshold policy, by using arrows to denote (depict) the state transitions caused by the cell retrieval prescribed by the policy. The leftheaded arrow denotes an LD retrieval whereas downheaded arrow denotes an LL retrieval.

Before defining the threshold policies mathematically, the following random vector

\(^6\)A policy which prescribes a single decision in each state, irrespective of the time or the sequence of previous decisions, is called a deterministic policy.
Figure 3.2: Graphic representation of the LDOLL threshold policy, \( R(\Delta) \), for \( Q = 4 \) and \( \Delta = 1 \). A single action corresponds to each state of the Markov chain.

\[
\begin{align*}
S(t) & \quad A(t) \\
N(t) & \quad N(t) - S(t) & \quad N(t + 1) \\
\downarrow & \quad \downarrow & \quad \downarrow \\
X & \quad Z & \quad Y
\end{align*}
\]

Figure 3.3: The temporal order of events and their corresponding random variables.

Variables are introduced in order to further simplify notation:

\[
\begin{align*}
X & \overset{\text{def}}{=} N(t) \\
Z & \overset{\text{def}}{=} N(t) - S(t) = X - S(t) \\
Y & \overset{\text{def}}{=} N(t + 1) \\
\end{align*}
\]

(3.12)

where \( X \) denotes the initial state, i.e. the state of the system at the begging of time slot \( t \), \( Z \) denotes the intermediate state, i.e. the state of the system after retrieval and before storage and \( Y \) denotes the ending state, i.e. the state of the system after both retrieval and storage have taken place. The time relation between the variables is presented on Figure 3.3.

Using the later variables, the LDOLL threshold policies, \( R(\Delta) \), are defined as follows:

\[
s(R(\Delta), x) = \begin{cases} 
(0, 0) & \text{if } x_{LL} = 0 \text{ and } x_{LD} = 0 \\
(1, 0) & \text{if } x_{LL} < \Delta \text{ and } x_{LD} > 0 \\
(0, 1) & \text{if } x_{LL} < \Delta \text{ and } x_{LD} = 0 \\
(1, 0) & \text{if } x_{LL} \geq \Delta , 
\end{cases}
\]

(3.13)

where \( \Delta \in \{1, \ldots, Q\} \) and \( s(R(\Delta), x) \) denotes the action prescribed by the policy in each state \( x \). Incrementing the threshold results in reduced expected LD cell delay and an
increased probability of LL cell loss, and vice versa. Thus with varying the threshold one can put more emphases on the objective of one of the classes.

The retrieval policy is the last necessary element for building a model of the LDOLL queue. First the one-step transition probability matrix of the model is defined. By noting that for each starting state \( x \), the intermediate state \( z \) is deterministically determined by the retrieval policy, the one-step transition probabilities can be constructed by simply adding the probabilities of the appropriate arrival patterns:

\[
P_{xy}(t) = \Pr\{N(t + 1) = y \mid N(t) = x\} = \sum_{a \in \Omega_{xy}} \Pr\{A(t) = a\},
\]

where \( \Omega_{xy} = \{a \in \Phi \mid \nu(x - s + a) = y\} \). The set \( \Omega_{xy} \) is the set of all arrival patterns that transfer the system from the intermediate state \( z = x - s \) to the ending state \( y \).

Because the threshold policies are deterministic it is easy to show that the sequence \( \{N(t)\} \) is a Markov chain. Therefore the state probability distribution of the chain at an arbitrary time instant \( t \), \( \Pr\{N(t) = y\} = \pi_y(t) \), can simply be found with the following recursive procedure:

\[
\pi_y(t) = \sum_{x \in \Xi} P_{xy}(t - 1) \cdot \pi_x(t - 1)
\]

\[
\pi_x(0) = I(x = x_0),
\]

where \( I(\cdot) \) is the indicator function and \( x_0 \) denotes the initial state of the queue.

For stationary input parameters, the LDOLL queue becomes an ergodic Markov chain. Every ergodic Markov chain will eventually reach equilibrium. The equilibrium state probabilities of the LDOLL queue are defined as: \( \pi_x \overset{\text{def}}{=} \lim_{t \to \infty} \Pr\{N(t) = x\} \), and can be obtained by solving the following system of equations:

\[
\sum_{x \in \Xi} \pi_x \cdot P_{xy} = \pi_x
\]

\[
\sum_{x \in \Xi} \pi_x = 1.
\]

Performance measures of interest

The performance measures that are considered in this subsection are the cell loss probability and the average cell delay of both LD and LL traffic classes. The next subsection also considers the cell delay variation.

The cell loss probability is discussed first. Let \( L(t) = (L_{LD}(t), L_{LL}(t)) \) denote the number of lost LD and LL cells respectively. One consequence of the LDOLL storage policy is that the LD cells can be lost as a result of either blocking of the newly arrived cells or replacement of cells already stored in the buffer, whereas LL cells can be lost only as a result of blocking. Hence the number of lost cells lost at time slot \( t \), can be expressed as:

\[
L(t) = Z + A(t) - Y.
\]
In other words, the number of LX cells lost at time slot $t$, equals the difference between the total number of cells in the CB and the IBs at time slot $t$ and the number of cells in the CB at the beginning at the beginning of the next time slot $t + 1$. Note that it always holds that $Y_{\text{LX}} \leq Z_{\text{LX}} + A_{\text{LX}}$ because in the case of blocking the number of actually stored cells is less than $A_{\text{LX}}(t)$, and in the case of replacement some of the previously stored cells are discarded, i.e. $Z_{\text{LD}}$ is reduced.

Using equation 3.17, one can calculate the expected number of lost cells by conditioning over all arrival patterns and all states:

$$E[L] = \sum_{x \in \mathbb{E}} \sum_{a \in \Phi} (x - s + a - \nu(x, s, a)) \cdot \zeta_1(a, t) \cdot \pi_x(t). \quad (3.18)$$

For the equilibrium analysis in this section, the time parameter can be omitted. Thus the LX loss probability can be found as:

$$\Pr\{\text{LX cell lost}\} = \frac{1}{\lambda_{\text{LX}}} \cdot \sum_{x \in \mathbb{E}} \sum_{a \in \Phi} (x - s + a - \nu(x, s, a)) \cdot \zeta_1(a) \cdot \pi_x, \quad (3.19)$$

where $\lambda_{\text{LX}}$ is obtained with (3.6) or (3.7), $\zeta_1(a)$ is calculated with (3.3) and $\pi_x$ is found by solving (3.16).

The second measure of interest is the average delay. One is interested in the delay of only those cells that eventually reach their destination. Thus by virtue of the Little's Law, the average cell delay can be expressed as:

$$E[W_{\text{LX}}] = \frac{E[N_{\text{LX}}]}{\lambda_{\text{LX}} - E[L_{\text{LX}}]}, \quad (3.20)$$

where the term in the denominator represents the fraction of cells which reach their destination and $E[N_{\text{LX}}]$ denotes the expected number of LX cells in the buffer. $E[N_{\text{LX}}]$ is easily found by $\sum_{x \in \mathbb{E}} x_{\text{LX}} \pi_x$ and the solution of (3.16).

Performance improvements of the LDOLL queue

This subsection investigates the performance improvements that can be achieved with the implementation of the LDOLL queue. It also investigates the impact of changing the threshold $\Delta$ and the cell mixing ratio $r_k$ on the buffer performance. The queue considered, has four inlets ($N = 4$), one outlet ($M = 1$), and a buffer size of ten ATM cells ($Q = 10$). In all experiments it is assumed that the traffic load is spread equally between the input links, $p_k = p$. The cell loss probability and the cell delay are found by equation (3.19) and (3.20), respectively.

The first set of graphs (see figure 3.4) compares the LDOLL and FIFO performance for balanced LD and LL traffic as input ($r_k = r = 0.5$). The graphs clearly illustrate the performance improvements of the LDOLL queuing. The LL loss probability is improved for approximately three orders of magnitude for the whole range of $p \in [0.4, 1.0]$, compared to the FIFO loss probability. Similarly the LD cells exhibit shorter delays, and correspondingly smaller delay variance, than the cells in the FIFO buffer. Thus the LDOLL queue yields significant performance improvements for the both traffic types.
3.1 Cell level analysis

Figure 3.4: Performance improvements of the LDOLL queue. Comparison of the loss (left graph) and delay (right graph) performance of the LDOLL and FIFO strategies as function of the load $N_p$. $Q = 10$, $\Delta = 5$, $N = 4$, $r_k = 0.5$.

Figure 3.5 illustrates the effect of changing the threshold $\Delta$ on the performance of the LDOLL queue. It is easily seen that increasing of $\Delta$ diminishes the performance improvement of the LL cells while enlarges the performance gains of the LD cells and vice versa. Note also that for very low threshold values (e.g. $\Delta = 2$), the LDOLL policy begins to perform unfairly, because for higher loads ($N_p > 0.5$), the LD cell delay becomes larger than the FIFO cell delay. Hence the LDOLL queue does not improve the performance of the both classes any more. Thus we can conclude that by changing the threshold $\Delta$ one can put more emphasis on the performance of one of the two traffic classes.

Figure 3.5: The effects of varying the threshold $\Delta$ on the performance improvements of the LD and LL traffic. $Q = 10$, $N = 4$, $r_k = 0.5$.

The last experiment considers the LDOLL queue under input traffic with varying mix of LD and LL cells. The results are depicted in figure 3.6. Observe that as $r \to 0$ the LL cell loss probability approaches the FIFO cell loss probability. This is because for $r \to 0$ there are almost no LD cell arrivals in the LDOLL queue. Hence the LDOLL queue behaves like a single class FIFO queue. The same effect is obtained as $r \to 1$, only this time the LD
cell loss probability approaches the FIFO cell loss probability. Similar conclusions can be
drawn from the delay graph. Note also that the performance of a traffic class improves as
it becomes a minority in the LDOLL buffer. Thus we can conclude that the LDOLL queue
performs best in a mixed traffic environment and for both limit cases, either LD traffic
only or LL traffic only, it behaves as a FIFO queue.

![Figure 3.6: Loss and delay performance for $N_p = 0.8$ and variable $r$. $Q = 10$, $N = 4$,
$M = 1$, $\Delta = 5$.](image)

In summary, the LDOLL queue improves the performance of both LD and LL traffic
classes significantly, and it achieves the highest gains in a mixed traffic environment. The
tradeoff between the performance of the two classes is controlled with the threshold $\Delta$.

### 3.1.3 Generalization of the LDOLL queue to multiple outlets

This subsection explores the applicability of the LDOLL principles when the queue has
multiple outlets ($M > 1$) sharing the same buffer resources (shared output buffering).
There are three basic questions which need to be answered: how should the LD and LL
cells be stored in the buffer, what is the optimal way to retrieve them, regarding their
respective QoS requirements, and how much will this extension improve the performance
of the LDOLL queue. Hence the subsection is build in a similar fashion. First, it describe
the generalized mathematical model and the storage policy, then the optimization of
the retrieval policy and finally the performance of the multiple outlet LDOLL queue in
equilibrium. The subsection contains only the most important results from the research.
More details can be found in [Sta94].

The multiple outlet LDOLL queue can be modeled as a multi-dimensional discrete time
(finite) Markov chain \(^7\). The multi-dimensionality comes from the fact, that one can
distinguish between $2M$ classes of cells in the CB, depending on their type (LD or LL)
and on the outlet they are destined to (1 ... $M$). Therefore the state of the queue, $N(t)$,

\(^7\)It is a Markov chain under certain assumptions about the retrieval policy, which will be discussed
later.
is now defined as:

\[ N(t) = (N_{LD1}(t), N_{LL1}(t), \ldots N_{LDM}(t), N_{LLM}(t)), \quad (3.21) \]

where \( N_{LXi} \) represents the total number of LX (LL or LD) cells in the buffer which are destined to outlet \( i \). The state space, \( \Xi \), then becomes:

\[ \Xi = \{ n \in \mathbb{N}^{2M} | n^T \cdot 1 \leq Q \}. \quad (3.22) \]

Obviously, the state space of such a queue is very large and it grows very fast with the increase of the buffer size \( Q \) and particularly with the increase of \( M \) (the number of outlets), namely:

\[ |\Xi| = \binom{Q + 2M}{2M} \quad (3.23) \]

Therefore I consider a rather simple case, with \( Q = 4 \), \( M = 2 \) and an arbitrary number of input links. This choice results in a queue with 70 states and four classes of customers, namely:

1. LD1 – cells destined to outlet 1 of type LD
2. LL1 – cells destined to outlet 1 of type LL
3. LD2 – cells destined to outlet 2 of type LD
4. LL2 – cells destined to outlet 2 of type LL

Note that this is the simplest multiple outlet LDOLL queue which possesses at least one state with four degrees of freedom, i.e. a state where the CS has a choice out of maximum number of actions (four in this case and 2\( M \) in general), namely for each outlet it can retrieve either an LD or an LL cell. Therefore the drawn conclusions are applicable to more general cases than \( M = 2 \).

**The Arrival process**

The arrival process at each time slot \( t, t \in \mathbb{N} \), is again described with the random vector variable \( A(t) \). Since the considered LDOLL queue has four classes of customers \( A(t) \) is extended to:

\[ A(t) = (A_{LD1}(t), A_{LL1}(t), A_{LD2}(t), A_{LL2}(t)), \]

where \( A_{LXi}(t) \) designates the total number of LX cells destined to output \( i \), at time slot \( t \). The set of all possible arrival vectors \( \Phi \) satisfies:

\[ \Phi = \{ a \in \mathbb{N}^4 | a^T \cdot 1 \leq N \}, \quad (3.24) \]

and has cardinality \( |\Phi| = \binom{N + 4}{4} \).

Using the same assumptions as in the single outlet case, the arrival process is modeled as Bernoulli process. Because of the multi-dimensionality, the Bernoulli process now modeled with three parameters:
$p_k(t)$ – the loading, or the probability of cell arrival at $IB_k$, during the time slot $t$.

$r_k(t)$ – the cell mixing ratio, or the probability that the arriving cell is of LD type.

$l_k(t)$ – the link mixing ratio, or the probability that the arriving cell at $IB_k$, is destined to output link $t$.

With these three parameters and assuming independence between the input cell streams, the joint probability distribution of $A_{LD1}(t)$, $A_{LL1}(t)$, $A_{LD2}(t)$ and $A_{LL2}(t)$ can be found again by $Pr\{A(t) = a\} = \zeta_1(a, t)$, where the recursive function $\zeta_k((i,j,m,n),t)$ is defined as the probability that $i$ LD1 cells, $j$ LL1 cells, $m$ LD2 cells and $n$ LL2 cells are present at $IB_{k...N}$ at time slot $t$. The recursive procedure is easily obtained from equation (3.3), see [Sta94].

In similar fashion, the mean arrival rate and the variance of the number of arrived cells, could be easily found by expanding equations (3.6) through (3.9).

The multi-outlet storage policy

Since the LDOLL queue considered has four classes of customers, the cells in the buffer are now organized in four linked lists. Each list keeps track of the order in which the cells from the respective class have arrived. The state of the queue at time slot $t$, is limited to four dimensions

$$N(t) = (N_{LD1}(t), N_{LL1}(t), N_{LD2}(t), N_{LL2}(t)).$$

The state space, $\Xi$, becomes four dimensional too, with cardinality

$$|\Xi| = (Q/4 + 1)(Q/3 + 1)(Q/2 + 1)(Q + 1).$$

At the beginning of every time slot, the CS retrieves cells from the CB according to some retrieval policy, described with the retrieval vector $S(t)$:

$$S(t) = (S_{LD1}(t), S_{LL1}(t), S_{LD2}(t), S_{LL2}(t)),$$

which represents the total number of cells of each class, retrieved at time slot $t$. After retrieval, the storage of the newly arrived ATM cells takes place.

In the single outlet case the storage problem is almost trivial. As discussed in the previous subsection, the obvious optimal storage policy is to give storage and replacement priority to the LL cells. Unfortunately this rather simple storage policy can not be easily expanded to the multi-dimensional case, because in the multiple outlet case the interrelationships of cells destined to different outlets, have to be resolved. For instance, may an arriving LL2 cell replace a previously stored LD1 cell in the buffer? Or if two LL cells destined to distinct outlets, are contending for a single buffer place, which one should be stored?

In order to resolve the apparent ambiguity, the optimal storage policy is required to satisfy the following three requirements:

In the $M > 2$ case, this parameters becomes $l_k(t)$ denoting the probability that the cell arriving at $IB_k$ is destined to outlet $t$. Naturally in that case, $\sum_{i=1}^{M} l_{ki} = 1$ must hold.
3.1 Cell level analysis

Optimality: it should result in the lowest possible loss probability for the LL cells and the smallest possible average cell delay and delay variation for the LD cells.

Generality: it should be a generalization of the single-outlet storage policy and in the same time it should be easily expandable to $M > 2$ cases.

Order independence: it should be independent of the order in which the IBs are polled, i.e. independent of the order in which the various types of cells, arriving in the same time slot, are transferred from the IBs to the CB.

The rationale behind the first two requirements is obvious. The third requirement guarantees implementational simplicity, because if the storage policy is order dependent, then the CE would have to poll all of the IBs before transferring the first cell, which is a time consuming operation. Additionally it also greatly simplifies the analysis since the enqueuing order does not have to be included in the model.

The storage policy found to satisfy all of three requirements is quite complex. It makes decisions based on the following hierarchy of principles:

1. Work conserving: all cells are stored as long as there is free buffer space. The principle stalls of course when the buffer is full.

2. LL cell priority: LL cells have push-out priority over LD cells, regardless of their destination. The principle stalls when there are multiple classes of LD cells or when there are no LD cells present in the buffer.

3. Output group balance: the cells of the minority cell group$^9$ get storage and replacement priority over the cells of the majority group$^{10}$. For instance, if the total number of stored LL1 and LD1 cells in a full buffer, is bigger than the total number of LL2 and LD2 cells, then a newly arrived LL2 cell will replace one of the previously stored LD1 cells. By the same principle an LL2 cell can overwrite an LL1 as long as $N_{LL1} > N_{LL2}$ and there are no LD cells left in the buffer.

4. Random choice: If non of the above principles yields an unambiguous decision then the CE chooses randomly from the set of the permissible actions.

The last principle is added because in $M > 2$ cases, certain situations arise when it is impossible to make an unambiguous decision. This is illustrated with the following example.

Say, that the LDOLL queue has eight buffer spaces ($Q = 8$) and four outlets ($M = 4$) and hence eight classes of customers, LD1...4 and LL1...4. Let's assume that at time slot $t$, after retrieval the buffer contains three LL1 and three LL4 cells. It is further assumed that there are three cells in the IBs waiting to be stored, one LL2 and two LL3. There are two empty buffer spaces left. There are various orders in which the cells can be stored, but lets assume that first the LL2 cell and one of the LL3 cells are stored. The remaining LL3 cell, according to the third principle, has replacement priority over both LL1 and LL4

---

$^9$A cell group is defined as the set of all cells destined to the same outlet.

$^{10}$Note however, that the second principle never allows favoring LD over LL cells.
cells, but the question is which of the two should it replace? There is no way to break this tie without assigning priorities to cells from different output streams. Note also that this ambiguity does not occur as a consequence of the LDOLL queuing, but it is inherent to the shared output buffering method.

The random choice was chosen as a fair priority mechanism to solve situations like the one just described and still comply with the third requirement for the optimal storage policy. Note that although the ending state \( Y \) is not uniquely determined, it does not depend on the order in which the IBSs are polled. Nevertheless in the two outlet case \( (M = 2) \), the CE is guaranteed to always be able to make a decision based on the first three principles only, i.e. the ambiguity never arises.

The state evolution equations in the \( M = 2 \) case are significantly more complicated than in the \( M = 1 \) case. The auxiliary random vector variables, \( X \), \( Y \) and \( Z \) (see equation (3.12) for definition), are used again in order to simplify notation. The equations can be written as follows:

\[
\begin{align*}
\text{yll}_1 &= \begin{cases} 
\min( Q - z_{ll}2 - A_{ll}2(t), \ z_{ll}1 + A_{ll}1(t)) \text{ if } z_{ll}2 + A_{ll}2 \leq Q/2 \\
\min( Q/2, \ z_{ll}1 + A_{ll}1(t)) \text{ otherwise .}
\end{cases} 
\tag{3.25}
\end{align*}
\]

\[
\begin{align*}
\text{yld}_1 &= \begin{cases} 
\min( Q_{ld} - z_{ld}2 - A_{ld}2(t), \ z_{ld}1 + A_{ld}1(t)) \text{ if } y_{ll}1 \geq Q/2 \\
\min( Q_{ld}, \ z_{ld}1 + A_{ld}1(t)) \text{ if } y_{ll}2 \geq Q/2 \\
\min( Q/2 - y_{ll}1, \ z_{ld}1 + A_{ld}1(t)) \text{ if } A = \text{TRUE} \\
\min( Q_{ld} - z_{ld}2 - A_{ld}2(t), \ z_{ld}1 + A_{ld}1(t)) \text{ otherwise .}
\end{cases} 
\tag{3.26}
\end{align*}
\]

where the condition \( A \) is defined as the joint event:

\[
A \overset{\text{def}}{=} y_{ll}1 \leq Q/2 \land y_{ll}2 \leq Q/2 \land z_{ld}2 + A_{ld}2(t) \geq Q/2 - y_{ll}2
\]

and

\[
\min(x,y) \overset{\text{def}}{=} (\min (x,y))^+
\]

\[
Q_{ld} \overset{\text{def}}{=} Q - y_{ll}1 - y_{ll}2.
\]

The formulas for \( y_{ll}2 \) and \( y_{ld}2 \) can be easily obtained from equations (3.25) and (3.26), by lexicographic substitution of 2 for 1.

The multi-outlet retrieval policy

Again only work conserving retrieval policies are considered as candidates. Thus in every time slot \( t \), the CS of the two outlet switch will retrieve at most two cells, one of group 1 and one of group 2. Depending on the state of the buffer \( N(t) \), the CS has a choice of one, two or even four different actions in some states. The set of all possible actions in
state $x$ is denoted with $K(x)$ and it is defined as:

$$K(x) = \begin{cases} 
\{(0, 0, 0, 0)\} & \text{if } x = 0 \\
\{(0, 0, 0, 1)\} & \text{if } x = (0, 0, 0, x_4) \\
\{(0, 0, 1, 0)\} & \text{if } x = (0, 0, x_3, 0) \\
\{(0, 1, 0, 0)\} & \text{if } x = (0, x_2, 0, 0) \\
\{(1, 0, 0, 0)\} & \text{if } x = (x_1, 0, 0, 0) \\
\{(1, 0, 1, 0)\} & \text{if } x = (x_1, 0, x_3, 0) \\
\{(1, 0, 0, 1)\} & \text{if } x = (x_1, 0, 0, x_4) \\
\{(0, 1, 0, 0)\} & \text{if } x = (0, x_2, 0, x_4) \\
\{(0, 0, 0, 1); (0, 0, 1, 0)\} & \text{if } x = (0, 0, x_3, x_4) \\
\{(0, 1, 0, 0); (1, 0, 0, 0)\} & \text{if } x = (x_1, x_2, 0, 0) \\
\{(0, 1, 0, 1); (0, 1, 1, 0)\} & \text{if } x = (0, x_2, x_3, x_4) \\
\{(1, 0, 0, 1); (0, 1, 1, 0)\} & \text{if } x = (x_1, 0, x_3, x_4) \\
\{(1, 0, 0, 1); (1, 0, 1, 0)\} & \text{if } x = (x_1, 0, 0, x_4) \\
\{(1, 0, 0, 1); (0, 1, 0, 1)\} & \text{if } x = (x_1, x_2, 0, x_4) \\
\{(0, 1, 1, 0); (0, 1, 1, 0)\} & \text{if } x = (x_1, x_2, x_3, 0) \\
\{(0, 1, 0, 1); (0, 1, 0, 1)\} & \text{if } x = (x_1, x_2, x_3, x_4) \\
\end{cases} \quad (3.27)$$

where $x_1 \ldots x_4 \in \mathbb{N} \setminus \{0\}$ and $x^T \cdot 1 \leq Q$.

The main objective is to find the retrieval policy that minimizes both the LL cell loss probability and the LD cell delay variance. Generally speaking, the retrieval policy can be time, state and history dependent, and in addition it may also prescribe more than one action in a single state, in which case a random mechanism determines which action is taken. Thus a policy $\mathcal{R}$ can be defined as a set of functions:

$$\mathcal{R} = \{ \mu(x, s, h, t) | t = 0, 1, \ldots \} \quad (3.28)$$

$$\mu(x, s, h, t) = \Pr\{ S(t) = s | N(t) = x, H(t - 1) = h \},$$

where:

$$\begin{cases} 
x \in \Xi \\
s \in K(x) \\
\mathcal{H}(t) = \{ N(t), S(t) \} \cup \mathcal{H}(t - 1) \\
\mathcal{H}(0) = \emptyset \\
t \in \{ 0, 1, \ldots \} 
\end{cases} \quad (3.29)$$

and $\mu$ satisfying $0 \leq \mu(x, s, h, t) \leq 1$, and $\sum_{x \in K(x)} \mu(x, s, h, t) = 1$.

Assuming that the arrival process is stationary and the retrieval policy, $\mathcal{R}$, is deterministic and time independent, which is shown later, the stochastic sequence $\{N(t)\}$ becomes an ergodic Markov chain with equilibrium state distribution given with the following set of equations:

$$\sum_{x \in \Xi} \pi_x(\mathcal{R}) \cdot P_{xy}(\mathcal{R}) = \pi_x(\mathcal{R})$$

$$\sum_{x \in \Xi} \pi_x(\mathcal{R}) = 1. \quad (3.30)$$
where \( \pi_x(\mathcal{R}) \) is defined as the limiting probability \( \pi_x(\mathcal{R}) \overset{\text{def}}{=} \lim_{t \to \infty} \Pr_T \{ N(t) = x \} \), and the one step transition probability matrix, \( P_{xy}(\mathcal{R}) \), is found using a similar approach as in the single outlet case (see equation (3.14)).

**Optimization of the retrieval policy**

In order to find the optimal retrieval policy from the infinite set defined in the previous subsection, I use the theory of Markovian decision processes. A Markovian decision process is defined as the stochastic process that describes the evolution of a dynamic system controlled by a sequence of actions. The evolution of the system is determined with the interaction between the laws of motion of the system and the sequence of actions taken over some period of time. The ultimate goal is to make those decisions which control the system in the optimal manner. In the case considered here the dynamic system is the LDOLL queue. The laws of motion are determined by the arrival process and the storage policy, and the sequence of actions corresponds to the sequence of decisions prescribed by the retrieval policy \( \mathcal{R} \). Thus the sequence \( \{ N(t), S(t), \ldots \} \) is a Markovian decision process.

The Markov decision theory offers several criteria for optimization. Here I employ the expected average cost per unit of time criterion. Given the initial state \( N(t = 0) = x_0 \), the criterion is defined with the following expression:

\[
\Psi(\mathcal{R}, x_0) = \lim_{T \to \infty} \frac{1}{T + 1} \cdot \sum_{t=0}^{T} \sum_{x \in \Xi} \pi_x(\mathcal{R}, t) \sum_{s \in \mathcal{R}(x)} c(x, s) \cdot \mu(x, s, h, t)
\]  

(3.31)

where the cost function, \( c(x, s) \), denotes the cost incurred when the process is in state \( x \) and action \( s \) is taken, and the objective function, \( \Psi(\mathcal{R}, x_0) \), represents the expected average cost per unit of time of operating the system under policy \( \mathcal{R} \).

In Markov decision theory it is proven (see [Der70, p.26]) that in order to minimize \( \Psi \), we do not have to consider the infinite number of policies defined by (3.28). The policy which minimizes the expected average cost per unit of time, belongs to a class of deterministic policies.

Thus, driven by a deterministic retrieval policy the LDOLL queue becomes an ergodic Markov chain and its state probability distribution, assuming stationary input process, becomes the equilibrium state distribution given with the set of equations (3.30). For any deterministic retrieval policy, the expected average cost per unit of time criterion reduces to:

\[
\Psi(\mathcal{R}) = \sum_{x \in \Xi} \pi_x(\mathcal{R}) \cdot c(x, s(\mathcal{R}, x)),
\]

(3.32)

where \( s(\mathcal{R}, x) \) denotes the decision made under policy \( \mathcal{R} \), when the queue is in state \( x \). Note that \( \Psi(\mathcal{R}) \) is a linear function of the equilibrium state probabilities, independent of time.
3.1 Cell level analysis

As stated earlier the objective of the LDOLL mechanism is to minimize the LL cell loss probability and the LD cell delay and delay variance. The LXi loss probability is given by:

$$\Pr_R\{\text{LX}_i \text{ cell loss}\} = \frac{E_R[L_{iX_i}]}{\lambda_{LX_i}}, \quad (3.33)$$

where the expected number of lost cells can be found again by (3.18), and $\lambda_{LX_i}$ by the expanded version of (3.7). Under the equilibrium assumption, the $E_R[L_{iX_i}]$ can also be found with a much simpler expression:

$$E_R[L_{iX_i}] = \lambda_{LX_i} - \sum_{x \in \Xi} \mu(x, (I(LX_i = LD_1), \ldots, I(LX_i = LL_2))) \cdot \pi_x(R). \quad (3.34)$$

where the indicator function $I(\cdot)$ is 1, only in states where the policy prescribes retrieval of the corresponding type of cells. Equation (3.34) is derived from equation (3.17), by substituting $E_R[X] = E_R[Y]$ and $E_R[Z] = E_R[X] - E_R[S(t)]$. The optimization procedure uses (3.34), note however that (3.18) is numerically more stable and it is also applicable in non-stationary regimes.

The average LXi delay can be calculated by equation (3.20). However, for the optimization purposes, I use the first approximation of equation (3.20), namely:

$$E_R[W_{iX_i}] = \frac{E_R[N_{iX_i}]}{\lambda_{LX_i}}, \quad (3.35)$$

where the term in the denominator neglects the fraction of cells that are lost along the way. This is necessary because the left hand side of equation (3.20) is not a linear function of the equilibrium state probabilities.

There is no exact formula for calculating the delay variance because of the complexity of the LDOLL queue. Therefore I employ an approximation which was derived in [Awa93]:

$$\text{Var}_R[W_{iX_i}] \approx \frac{\text{Var}_R[N_{iX_i}] - \text{Var}_R[A_{LX_i}] \cdot E_R[W_{iX_i}]}{\lambda_{LX_i}^2}, \quad (3.36)$$

where $\text{Var}_R[A_{LX_i}]$ can be obtained from the expanded versions of equations (3.8) through (3.9), and the $\text{Var}_R[N_{iX_i}]$ is found in the standard way from the equilibrium state probabilities.

The approximation is valid per cell group, because it was derived assuming that, at time slot $t$, only one cell leaves the system.

The next step is to express the objective function in terms of the performance measures. Unfortunately equation (3.36) is a non-linear function of the equilibrium state probabilities and Markov decision theory is not suited for minimizing objective functions which are non-linear functions of the state probabilities. Hence equation (3.36) can not be included in $\Psi$.

Therefore I proceed with minimization of the LL cell loss probability and the average LD cell delay only, assuming that minimizing the LD cell delay will also reduce the LD delay variance. This heuristic approach was tested by calculating the LD delay and delay variance performance of a large number of arbitrary deterministic policies. The results are
shown on the "delay-variance" plane in Figure 3.7. Every dot represents the performance of an arbitrary deterministic policy whereas the triangles depict the performance of the optimal policies (to be presented next).

The figure clearly shows the validity of the hypotheses. This can be seen from the fact that all optimal policies lie very close to the diagonal of the "delay-variance" plane, which means that the cell delay variance is indeed reduced by minimizing the average cell delay. The same phenomenon was also observed in the single outlet case.

![Figure 3.7: The relationship between the mean LD delay and the LD delay variance for M = 2, Q = 4, and 2048 deterministic policies](image)

In order to optimize the LL cell loss probability and average LD cell delay simultaneously, the objective function is build as a linear combination of both objectives, namely:

\[
\Psi(R) = \alpha \frac{\lambda_{LD} E_{R}[W_{LD1}] + \lambda_{LD2} E_{R}[W_{LD2}]}{\lambda_{LD}} + \beta \frac{\lambda_{LL1} P_{LR}\{LL1 \text{ cell lost}\} + \lambda_{LL2} P_{LR}\{LL2 \text{ cell lost}\}}{\lambda_{LL}}.
\]  

(3.37)

Note that the LXi factors cancel out against the denominators of (3.33) and (3.35). Thus by employing equation (3.34) and omitting the constant part of the LL loss probability,

\footnote{The graph is called the "delay-variance" plane because the x-coordinate represents the average LD delay and the y-coordinate the LD delay variance, under policy R.}
the cost function is defined as follows:

\[
c(x, s) = \begin{cases} 
0 & \text{if } x = 0 \\
\alpha \frac{\pi_{LD1} + \pi_{LD2}}{\lambda_D} & \text{if } x \neq 0 \text{ and } S_{LL} = 0 \\
\alpha \frac{\pi_{LD1} + \pi_{LD2}}{\lambda_D} - \frac{\beta}{\lambda_L} & \text{if } x \neq 0 \text{ and } S_{LL} = 1 \\
\alpha \frac{\pi_{LD1} + \pi_{LD2}}{\lambda_D} - \frac{2\beta}{\lambda_L} & \text{if } x \neq 0 \text{ and } S_{LL} = 2 
\end{cases}
\]  
(3.38)

where \( S_{LL} \) is defined as: \( S_{LL} = S_{LL1} + S_{LL2} \). By changing the coefficients \( \alpha \) and \( \beta \) one can put more emphasis on one of the two objectives.

Even though the set of possible retrieval policies was remarkably reduced, the set of deterministic policies is still very large. For the LDOLL queue with \( Q = 4 \) and \( M = 2 \) the set consists of \( 2^{30} \) deterministic policies. To find the optimal one in this large set, I used a special algorithm called the value iteration algorithm. It is very suitable for queues with large state spaces because it performs the optimization without solving the set of equilibrium state equations at each iteration step. A description of the algorithm can be found in [Tij86].

Multiple outlet threshold policies

Before presenting the results of the optimization, I introduce two special types of retrieval policies, which are generalizations of the single outlet LDOLL threshold policies. The first type is a straight-forward generalization of the threshold policies in the single outlet case, called the myopic threshold policy. It makes decisions considering the cells of each outlet group separately. In other words, which cell will be forwarded to the OB, depends only on the number of LL, cells in the buffer. Effectively the policy is blind for cells belonging to groups other their own, hence its name.

By introducing the auxiliary retrieval vector \( S_i = (S_{LD1}, S_{LL}) \), the myopic threshold policy is defined as:

\[
S_1(\mathcal{R}(\Delta), x) = \begin{cases} 
(0, 0) & \text{if } x_{LD1} = 0 \text{ and } x_{LL1} = 0 \\
(1, 0) & \text{if } x_{LL1} < \Delta \text{ and } x_{LD1} > 0 \\
(0, 1) & \text{otherwise} 
\end{cases}
\]  
(3.39)

and \( S_2(\mathcal{R}(\Delta), x) \) is obtained with a lexicographic substitution of 1 with 2.

The second type is called the joint threshold policy. The name reflects the fact that the joint threshold policy prescribes decisions by comparing the threshold \( \Delta \) with the total number of LL cells \( (x_{LL} = x_{LL1} + x_{LL2}) \) present in the CB. It is defined as:

\[
S_1(\mathcal{R}(\Delta), x) = \begin{cases} 
(0, 0) & \text{if } x_{LD1} = 0 \text{ and } x_{LL1} = 0 \\
(1, 0) & \text{if } x_{LD1} > 0 \text{ and } (x_{LL1} = 0 \lor x_{LL} < \Delta) \\
(0, 1) & \text{otherwise} 
\end{cases}
\]  
(3.40)

and \( S_2(\mathcal{R}(\Delta), x) \) can again be obtained with a lexicographic substitution of 1 with 2.
In order to illustrate the structure of the threshold policies, a myopic threshold policy with $\Delta = 2$ is presented on figure 3.8. The four-dimensional state space is presented as a two-dimensional picture with the following method. First, by fixing $x_{LL_1}$ the state space is split into $Q + 1$ three-dimensional slices, and positioned on top of each other. Then each three-dimensional slice is cut into several two-dimensional triangle subspaces by fixing $x_{LD_1}$. Finally all triangle subspaces are arranged horizontally, left to right in a decreasing size order. Note that the method can be easily extended to higher dimensions, by fixing the values various groups sequentially.

![Diagram](image)

Figure 3.8: The structure of a myopic threshold policy with $\Delta = 2$ for an LDOLL queue with $Q = 4$ and $M = 2$.

In each state, the small black arrow denotes an LX2 retrieval and the large gray arrow denotes an LX1 retrieval. In both cases, a down-headed arrow means that an LL cell is retrieved, a left-headed arrow means that an LD cell is retrieved and a square represents the action "do nothing". For instance, the symbol consisting of a black square and a left-headed gray arrow means that in that state the myopic threshold policy prescribes that only one LD1 cell should be retrieved.

Note that both policies reduce to the LDOLL threshold policies for the $M = 1$ case. One can think of other threshold policies by defining different dependencies between the cells of different classes, but these two are sufficient for comparison with the optimal policies that result from the optimization process.
3.1 Cell level analysis

The optimal retrieval policies

The previously described optimization procedure produced a set of six optimal policies, obtained for the whole range of values for the coefficients $\alpha$ and $\beta$. The performance of these policies is compared with the performance of a large number of arbitrary deterministic policies and the two types of threshold policies, defined in the previous subsection. The result is presented in figure 3.9.

![Figure 3.9: Objective space for two output LDOLL queue with 2048 randomly chosen deterministic policies, myopic and the joint threshold and optimal policies. $Q = 4$; $r_1 = 0.4$, $r_2 = 0.3$, $r_3 = 0.2$, $r_4 = 0.1$; $p_1 = 0.4$, $p_2 = 0.2$, $p_3 = 0.1$, $p_4 = 0.05$; $l_{1...5} = 0.5$.](image)

The figure actually represents the two-dimensional objective space of the optimization, where the abscissa denotes the average LD cell delay and the ordinate denotes the LL cell loss probability. Hence the graph is called the "loss-delay" plane. The policy has an optimal position in this space if no other policy can produce lower mean delay for the LD cells (further west) and at the same time, lower cell loss probability for the LL cells (further south). Each dot in the figure is obtained with applying a different retrieval policy and calculating both performance measures. The figure contains the optimal, the myopic and joint threshold policies and 2048 randomly chosen retrieval policies, which are used to illustrate the gains of the optimization more clearly.

Two important conclusions can be drawn from the "loss-delay" plain. First, note that the points corresponding to the threshold policies, lie very close to the lower (south-west)
convex hull of the mapping of the solution set onto the objective space\textsuperscript{12}. In other words the threshold policies are performing almost as good as the optimal policies. Second, the myopic threshold policies are optimal for all values of $\Delta$, and a wide range of the input parameters.

The analysis has also shown that the obtained optimal solutions are very much dependent on variations in the loading of the LDOLL queue, contrary to the single outlet case where the optimal policies are independent of variations in the input traffic parameters.

Figure 3.10 contains the loss-delay plane obtained by applying the value iteration algorithm to an LDOLL queue with $Q = 6$. The algorithm produced a new set of optimal solutions. Note first that the number of optimal policies remarkably increases to more than 16. Second, some of the optimal policies have almost the same performance in spite of their distinct structure. This can be seen from figure 3.10, where fourteen of the optimal policies are presented and one can hardly distinguish between more than ten of them. Third and most important, the myopic threshold policies are not optimal for all values of the threshold $\Delta$. The author was not able to find any combination of input conditions which would result in an optimal solution which covers all of the myopic poli-

\textsuperscript{12}In the single-outlet case the threshold policies are the vertices of the hull.
3.1 Cell level analysis

cies. Nevertheless their performance is still very close to the performance of the optimal policies.

The excellent performance of the myopic threshold policies is very important for two reasons. From a generalization point of view, the myopic threshold policies are very attractive because they can be directly extended to any higher dimension \( M > 2 \), whereas the optimal policies are very difficult to define in cases with more than two outlets. From the implementation point of view, the myopic threshold policies, with their simple structure are much more desirable than the highly complex optimal policies. This implementation simplicity combined with the excellent performance, makes the myopic threshold policies the best practical solution for the multiple-outlet LDOLL queue.

Performance of the multi-outlet LDOLL queue

This subsection investigates the performance of the two-outlet LDOLL queue controlled by a myopic threshold policy. In all three experiments, the buffer size is four \( (Q = 4) \), the threshold of the myopic policy is two \( (\Delta = 2) \) and the queue is fed with a symmetric load \( (r_k = r, \ l_k = l \text{ and } p_k = p \ \forall k \in \{1, \ldots N\}) \), except in the last experiment which explores the impact of the unbalanced loading on the performance of the two-outlet LDOLL queue.

The \( L_X \) loss probability is found as:

\[
Pr\{L_X \text{ cell loss}\} = \frac{E[L_{LX1}] + E[L_{LX2}]}{\lambda_{LX1} + \lambda_{LX2}} \tag{3.41}
\]

where \( E[L_{LX1}] \) is obtained from equation (3.33). The \( L_X \) mean delay is obtained from equation (3.35). The average \( L_X \) cell delay is found by summing of the mean delays of all classes.

The three experiments are: variable loading, variable cell mixing ratio and unbalanced link loading. The first experiment investigates the behavior of the cell loss probability and the mean cell delay, when the load of the queue is varied from 0.2 till 0.9 and the queue is fed with a balanced mixture of cells of the two classes, \( r = 0.5 \) and \( l = 0.5 \). Figure 3.11 compares both measures with the performance of a two-outlet LDOLL queue with separate output buffering, and a shared output FIFO queue.

The graphs clearly show the benefits of the LDOLL queuing. Note from the first graph that the average LD cell delay, in both separate and shared buffering cases, is smaller than the average cell delay in the shared FIFO buffer, which is in turn smaller than the average LL cell delay. As the load increases, the difference becomes more significant. Obviously the LL cells must get something in return for the larger mean delays. Hence, the LL cells have much lower cell loss probability than the LD cells.

The graphs also show the benefits of the shared output buffering with respect to the separate output buffering. One can observe from the delay graph that with a same buffer size the shared output buffering achieves lower cell loss probabilities than the separate output buffering. This is traded for a slight increase of the mean cell delay, as can be seen from the loss graph. Nevertheless the mean LD cell delay for the two outlet case is still significantly smaller than the mean LL cell delay in the single outlet case. On the other
Figure 3.11: Loss and delay versus loading for $Q = 4$, $\tau = l = 0.5$ and $\Delta = 2$.

hand, the LL loss probability in the single outlet case is larger than the LD loss probability in the two outlet case, so it is clear that the shared output buffering improves the overall performance of the LDOLL queue. The observed gains are expected to increase for larger $Q$s.

The second experiment considers the performance of the both measures, for varying cell mixing ratio, fixed loading $N_p = 0.7$ and fixed link mixing ratio $l = 0.5$. Both graphs are given in figure 3.12.

As can be seen from the curves, the two-outlet LDOLL queue converges to the two-outlet
3.1 Cell level analysis

Figure 3.12: Loss and delay as a function of $r$ for $l = 0.5$, $Q = 4$, $\Delta = 2$ and $N_p = 0.7$

shared FIFO queue, for cell mixing ratio values close to either one or zero ($r \to 0$ or $r \to 1$). This is not surprising since for the extreme values of $r$ the LDOLL queue contains cells of only one type, LD or LL, respectively. Therefore the LL cell loss probability approaches the FIFO loss probability as $r \to 0$ and LD cell loss probability approaches the FIFO loss performance as $r \to 1$. The same is true for the average LD and LL cell delay, respectively.

Therefore the conclusions drawn in the $M = 1$ case, remain to be valid for the two-outlet case. Namely, the measure of performance of the corresponding class improves as it becomes a minority in the queue. This can be seen on the example of the mean LD cell delay which decreases from 2.3 for $r = 0.8$, to 2.15 for $r = 0.2$. On the same range the LL loss probability falls from approximately $10^{-4}$ down to $10^{-6}$.

The third experiment investigates the behavior of the measures of interest for each class separately, when one of the outlets of the LDOLL queue is heavily loaded with LD cells and the other with LL cells. The load of outlet 1 consists of 90% LD cells, whereas the load of outlet 2 consists of 90% LL cells. The objective is to test whether the loss performance of the LD cells would be significantly deteriorated, in cases of asymmetric routing, which is very common in real networks.

The graphs in figure 3.13 show that this is not the case. The LD loss probability does not show any remarkable increase for both outlets compared to the balanced case. The LD2 cells have the largest loss probability. Because of the overwhelming number of LL2, they are more often replaced. Still, their loss probability is just slightly higher than the loss probability of LD1 cells. Similarly, the LL1 loss probability is a bit lower than the LL2 loss probability because on average, there are far less LL1 cells than LL2 cells in the buffer. The delay graph also does not show any significant changes compared to the balanced case. The LL1 mean delay is the largest because the much higher number of LD1 cells with their retrieval priority, forces the occasionally arrived LL1 cell to wait long.

One can conclude that LDOLL queuing with shared output buffering, is clearly beneficial even in cases when the output links of the switch are heavily loaded with cells of opposite types.
Figure 3.13: Loss and delay as a function of the loading for $r_1 = 0.9$ and $r_2 = 0.1$ and $Q = 4$, $l = 0.5$, $\Delta = 2$.

3.2 Burst level analysis

Previous section considered the LDOLL queue, under stationary conditions (constant arrival probabilities) at the CELL level. However the performance in non-stationary environment, where the arrival probabilities are dynamically changing at the higher level, is more relevant for practical implementations of the LDOLL strategy. Therefore this section is concerned with performance evaluation at the BURST level.

The objective is of the section is twofold. First, to determine the performance of the LDOLL queue in more realistic settings, in particular bursty traffic and larger buffer sizes (see subsection 3.2.2). In the previous section the arrival rate was constant and it never exceeded the output capacity of the system ($\sum_{k=1}^{N} p_k \leq 1$). Since the LDOLL queue provides statistical QoS guarantees, in this section the arrival rate varies in time and occasionally it exceeds the capacity of the outgoing link. Also the buffer sizes considered here are an order of magnitude larger than in the previous section.

The second objective is to determine the impact of the self-similarity property on the buffer performance. In particular, the performance of both classical and self-similar models, which were described in chapter 2, is compared to the performance of the actual traffic measurements.

The next subsection discusses various performance evaluation methods and describes simulation, the method which is used in the analysis.

3.2.1 Performance evaluation methods

There is an immense number of performance evaluation methods described in the literature (see [Man96] for an overview). I consider only methods that analyze the BURST and CELL level performance of a single ATM buffer. Interestingly, the available literature on methods analyzing both levels simultaneously, is significantly smaller than available literature on methods analyzing a single level only. In general one can distinguish between analytical and simulation methods. The analytical methods can be categorized into four categories (see [Man96]).
3.2 Burst level analysis

(I) **Exact methods** provide explicit results for the buffer contents and/or the delay distribution. Consequently the traffic models are kept relatively simple and are usually based on Markovian processes. Classical references for the cellular level analysis are [Kle75, Coh69]. A famous reference for the burst level analysis is [AMS82]. It considers an infinite buffer with constant output rate, loaded with a finite number of homogeneous Markov fluid sources. The buffer contents is given in terms of the eigenvalues and the eigenvectors of the system, which can be calculated explicitly. Unfortunately explicit results are available only for a very small number of traffic and system models.

(II) **Numerical methods** are used when the exact methods fail to produce explicit results. This is often the case when the solution is given as a set of equations or an eigensystem (eigenvalues and eigenvectors), which can not be solved explicitly. For instance in [Kos86], Kosten extends the results of [AMS82] by employing multiple classes of Markov fluid sources. The buffer contents is again expressed in terms of the eigenvalues and the eigenvectors of the system. Since they can not be found explicitly, he proposes a numerical procedure. Although many more models can be solved numerically than explicitly, the procedures involved are still very much dependent on the system under study and the employed traffic model.

(III) **Heuristic methods** are used when the exact and numerical methods can not produce a satisfactory result (solution). The basic principle is to approximate the difficult system with a system whose solution is available or easy to find. Several examples can be found in [Awa94] which further extends the fluid model of [AMS82] and [Kos86], by introducing an infinite buffer into the system. The solution of the finite buffer case is approximated by the solution of the infinite buffer case, in the spirit of [Tij92].

(IV) **Asymptotical methods** are often used instead of heuristics when no exact results are available. They yield asymptotic expansions, i.e. limiting results when one of the variables tends to infinity. In the ATM context, a very popular example, is the effective bandwidth concept which is a large buffer asymptotic. It is based on the assumption that, for a relatively large number of fluid models, the overflow probability of a buffer with size $B$, is asymptotically exponential, i.e.,

$$\Pr\{x > B\} \approx \alpha e^{-\eta B} \text{ as } B \to \infty,$$

where the constants $\eta > 0$ and $\alpha > 0$ are critically dependent on the employed traffic model. More details on the concept of effective bandwidth are given in chapter 4.

An alternative to the analytical methods is simulation. Simulation allows analysis of complex and large systems with arbitrary traffic models. The disadvantage is the very long execution time of standard simulation techniques involving rare events. Therefore various acceleration techniques have been developed. One of the most popular is the “importance sampling” (for examples see [Man96, AT96]). The basic principle is to modify the system in such a way that the rare events become frequent, and then to unbiash the obtained results by weighting with the relative likelihood of the observation in the modified system with respect to the original one. However these techniques are dependent on the system under study as well as the employed traffic model.
For the purpose of the analysis in this section the evaluation method should satisfy the following three requirements:

1. It should able to obtain very low probabilities in a reasonable time. Cell loss probabilities ranging from $10^{-6}$ to $10^{-12}$ are very typical for ATM applications.

2. It should be able to accommodate priority buffers, in order to analyze the \textsc{burst} level performance of the \textsc{ldoll} queue.

3. It should be able to incorporate various traffic models (both Markovian and non-Markovian), so that their performance can be compared properly.

It is obvious that none of the above methods satisfies all three requirements. The analytical methods have difficulties with the last two requirements. There is no explicit solution for the \textsc{ldoll} queue and numerical procedure used in the previous section is only valid for the Bernoulli traffic. There are no available heuristics at present and the asymptotical methods assume \textsc{fifo} buffers almost exclusively. The simulation method on the other hand satisfies both requirements 2 and 3, but has a difficulty with the first requirement. Importance sampling is a viable possibility, but each traffic model would require a separate implementation.

Therefore I chose a hybrid method called \textit{simputation}, developed by Awater and Schoute in [AS92]. It offers flexibility in modeling of the input traffic and the queuing system, and it is able to deal with events with arbitrarily small probabilities. The simputation method combines traffic simulation at the \textsc{burst} level with the queuing model computation (numerical calculation) at the \textsc{cell} level, hence the name. It retains the qualities of both methods. Thanks to the simulation part, an arbitrary traffic model can be employed at the burst level, while the computation part takes care of the low loss probabilities by keeping track of all possible realizations at the \textsc{cell} level (see the description below).

\section*{The simputation method}

This subsection provides a condensed description of the simputation method, tailored to the needs of the following subsections. A more general treatment can be found in [Awa94].

The simputation method assumes a Generally Modulated Compound Bernoulli Process (\textsc{gmcbp}) as an input traffic process. \textsc{gmcbp} means that the arrival process at the \textsc{cell} level, is modeled with a Compound Bernoulli process, whose parameters are generally modulated at the \textsc{burst} level. The modulation profile is provided by a simulation of the \textsc{burst} level traffic models described in subsection 2.3.2, but one can also use a measured traffic trace or an artificially constructed profile.

The modulation profile should provide the cell arrival rates of all input links. Mathematically, it is represented by a vector set \{$(\omega_i(t), \ t \in \{0,1,\ldots,T-1\}$\}, where $\omega_i(t)$ denotes the cell arrival rate on input link $i$, and the time slot $t$ corresponds to an ATM cell transmission time (i.e. \textsc{cell} level time unit). Since changes in the arrival rates $\omega_i(t)$, occur according to a \textsc{burst} level process, which is much slower than the \textsc{cell} level process, the arrival rates remain constant for large number of successive time slots. Therefore instead of specifying the arrival rate at each time slot, the modulation profile
could specify only the BURST level rate changes. Hence the modulation profile is reduced to \( \{ \omega_i(T_i) \}, T_j \subseteq \{0, 1, \ldots, T - 1\} \), where \( T_j \) denotes the time slot when a rate change occurs. This yields great reduction in computation effort as described later.

At cell level the method assumes that the queue can be modeled as a recurrent Markov chain. Let \( \Xi \) denote the state space of the Markov chain and \( x \) the state at time \( t \). The one-step transition probability matrix \( P_{xy}(t) \), can be easily obtained from the parameters of the Bernoulli process (e.g. see (3.14). The time dependence stems from dependence of the parameters on the BURST level variations. Additionally there may be other direct dependencies, inherent to the queuing model.

The evolution of the Markov chain in time, is described with the time-dependent state probability distribution \( \pi(t) \). Given the initial state distribution \( \pi(0) \), the state distribution at an arbitrary time instant \( t \), can be found with the following recursion:

\[
\pi(t) = \begin{cases} 
\pi(t - 1) \cdot P_{xy}(t - 1) & \text{for } 0 < t \leq T - 1 \\
\pi(0) & \text{for } t = 0 
\end{cases}
\] (3.42)

Thus the computation part of the simulation process needs only to update \( P_{xy}(t) \) and multiply it with the current state distribution vector \( \pi(t) \), at each time slot \( t \). The result is a set of probability distribution vectors \( \{ \pi(0), \pi(1), \ldots, \pi(T - 1) \} \), which summarize the statistics of all sample paths realizable under specific modulation profile. Since the updating of \( P(t)_{xy} \) is computationally the most expensive operation, it is clear that restricting the updates to only those time slots when a BURST level rate change occurs, achieves great savings in computation time. Other implementation issues that greatly reduce the computation time are discussed in [Awa94].

Note that the method can also accommodate multiple classes of customers, as long as the Markov chain remains numerically tractable. Note also that although various traffic models can only be applied at the BURST level, this is not seen as a restriction because BURST level variations are probably determinative for the buffer performance and furthermore, it is very difficult to measure the traffic at the CELL level, because of the high transmission speeds and the large amount of data that should be processed.

The main drawback of the simulation method is that it does not scale well with increase of the size of the state space \( |\Xi| \). Namely, the computation effort is proportional to \( |\Xi| \). For FIFO queue \( |\Xi| = O(Q) \), however for multi-class queues \( |\Xi| \) grows exponentially in \( Q \). Therefore for very large queues, heuristics maybe necessary at the CELL level, in order to reduce the long execution times.

The performance measures of interest can be obtained from the probability distribution vector set \( \pi(t) \). A detailed formal treatment can be found in [Awa94], here I derive only the results necessary for the analysis in the following subsections. We are interested in the overall performance metrics, in particular average delay and probability of loss over the whole stretch of time \( 0, \ldots, T - 1 \).

The overall loss probability can be found fairly easy. Let \( \text{Pr}\{\text{Loss}\}(t) \) denote the probability that a cell, arrived in time slot \( t \), will be lost immediately or in the future under any possible BURST level scenario. Then, the overall loss probability is found by weighting the \( \text{Pr}\{\text{Loss}\}(t) \) with the expected number of arrived cells at time slot \( t \), \( \lambda(t) \),
i.e.:

\[ \Pr\{\text{Loss}\} = \frac{\sum_{t=0}^{T-1} \lambda(t) \Pr\{\text{Loss}\}(t)}{\sum_{t=0}^{T-1} \lambda(t)} \]  

(3.43)

Since \( \lambda(t) \Pr\{\text{Loss}\}(t) = E[L(t)] \), where \( E[L(t)] \) denotes the expected number of lost cells arrived at time slot \( t \), the last equation reduces to:

\[ \Pr\{\text{Loss}\} = \frac{\sum_{t=0}^{T-1} E[L(t)]}{\sum_{t=0}^{T-1} \lambda(t)} \]  

(3.44)

For calculation of the overall average delay, I use the deterministic framework described in \[\text{Awa94}\]. Let \( N_r \) denote the accumulated cell waiting time in the system in realization \( r \) and \( E_r[W] \) denote the average cell delay in realization \( r \). Then the overall average delay is found by weighting \( E_r[W] \), with the product of the total number of retrieved cells in realization \( r \), \( S_r \), and the probability of occurrence of realization \( r \), \( p_r \):

\[ E[W] = \frac{\sum_{r \in R} E_r[W] \cdot S_r \cdot p_r}{\sum_{r \in R} S_r \cdot p_r} \]

\[ = \frac{\sum_{r \in R} N_r \cdot p_r}{\sum_{r \in R} S_r \cdot p_r} \]

\[ = \frac{E[N]}{E[S]} \]  

(3.45)

where the \( E_r[W] \cdot S_r = N_r \) follows from the Little’s Law, and \( E[N] \) and \( E[S] \) denote the expected number of cells present at, or retrieved from the buffer, respectively. Then the overall average delay is obtained by:

\[ E[W] = \frac{\sum_{t=0}^{T-1} E[N](t)}{\sum_{t=0}^{T-1} E[S](t)} \]  

(3.46)

Both \( E[N](t) \) and \( E[S](t) \) can be found from the computation result for \( \pi(t) \), \( t \in \{0, \ldots, T-1\} \) and the definition of the expected value.

Note that both equations are also applicable to multi-class systems to each customer type separately. Equations (3.44) and (3.46) will be used in the next subsection in order to calculate the desired performance measures.

### 3.2.2 Burst level analysis of the LDOLL queue

Before presenting the results of the analysis, I briefly discuss the queuing model, the traffic description and the evaluation method details, of the carried experiments.

The analyzed system is an single-outlet LDOLL queue with four input links, like the one described in subsection 3.1.2. Since I am interested in the performance under realistic conditions, the 10 minute actual traffic traces described in section 2.3.2, are used as input traffic. Three input links carry aggregated WWW traffic while the forth input link carries VBR video traffic corresponding to the digitized movie. In the LDOLL context, it
is assumed that the WWW streams contain only LL cells whereas the video trace contains only LD cells. The speed of all links, input and output, is assumed equal.

The model depicts an ATM multiplexer at the edge of a campus network, that combines traffic from several local sources on a single output link and sends it to the WAN edge switch. This agrees well with the measured traffic traces, because the WWW traces contain packet counts only for packets with destination address outside of the university campus network. This has also influenced the choice of the link speed. Recall that the measurements were made on a 10 Mbit/s backbone of the faculty. Since external WWW traffic is just a fraction of the traffic traveling through the LAN, all links in the model are set to T1 speed (1.5Mbit/s), as the closest typical link speed found on ports of commercial ATM switches (see e.g. specifications of Fore ATM 200BX/1000).

In order to apply the simulation method, the cell arrival probabilities at the CELL level must be calculated from the BURST level arrival rates. All traces contain the number of bytes transmitted in a unit interval. It is assumed that the bytes are packed into cells and then transmitted at a constant rate during the interval. Let $\omega_{t,k}(t)$ denote the instantaneous arrival rate at the BURST level, i.e. the number of bytes per $\Delta t_k$ seconds. The cell arrival probability at the CELL level, $p_k(t)$, is found as:

$$
p_k(t) = \frac{(\omega_{t,LD} + \omega_{t,LL}) \cdot 8}{C \cdot \Delta t_k},$$

(3.47)

where $C$ denotes the link speed in Mbits/s. The $p_k$s are kept constant for the duration of the interval $\Delta t_i$. Note that equation (3.47) neglects the ATM protocol overhead, and therefore slightly underestimates the actual cell arrival rate. That is not a concern because of the way the outgoing link utilization is varied.

The cell mixing ratio is obtained in a similar fashion, namely:

$$
r_k(t) = \frac{\omega_{t,LD}}{\omega_{t,LD} + \omega_{t,LL}}.

(3.48)

For the case considered here, the $r_k(t)$ is constant and known in advance, for $k = 4$, $r(k) = 1$ and $r(k) = 0$ otherwise. Thus the only variable parameter is $p_k(t)$.

The above method is applied to the WWW and the VBR video traces. For the WWW traffic, the $\Delta t$ is chosen equal to the measurement period of 10 ms. For the VBR video traffic the frame duration was set to 40 ms, in order to synchronize the two types of traces. Additionally the VBR video bandwidth was reduced by a factor 8, in order to achieve balanced LD and LL loading of the LDOLL queue.

Since the CELL level arrival probabilities are modulated according to equation (3.47), for a given modulation profile the mean loading of the LDOLL queue is fixed at a certain value. In order to achieve variable loading, the BURST level arrival rates are multiplied with a constant factor $\alpha$ for the whole range of time $t \in \{0, \ldots, T - 1\}$. The advantage of the method is that it keeps the correlation structure of the trace intact. Note that in cases when the mean load is increased ($\alpha > 1$), some of the arrival rates may exceed

\[^{13}\text{Thus playing the movie slightly faster, at the frame rate of 25 Hz instead of 24 Hz.}\]
the link capacity. The probability of such events\textsuperscript{14} however, is rather small. Might that happen, the arrival rate can be safely clipped, to the link capacity, without significantly changing the statistics of the process.

All necessary building blocks for the simputation of the single-outlet LDOLL queue were derived in the previous section. The state distribution at time $t$, $\pi(t)$, is found with the recursion procedure given by equation (3.15). The expected number of lost LX cells at time slot $t$, $E[L_{LX}(t)]$, is found by incorporating the $\pi(t)$ into equation (3.18), while $\lambda_{LD}(t)$ and $\lambda_{LL}(t)$ are found with (3.6) and (3.7), respectively. The $E[N_{LX}(t)]$ and $E[S_{LX}(t)]$ are found with the following formulas:

\begin{align}
E[N_{LX}(t)] &= \sum_{x \in \mathcal{S}} x_{LX} \cdot \pi_x(t) \tag{3.49} \\
E[S_{LX}(t)] &= \sum_{x \in \mathcal{S}} s_{LX}(R(\Delta), x) \cdot \pi_x(t), \tag{3.50}
\end{align}

where $s_{LX}(R(\Delta), x)$ denotes the action prescribed by the threshold policy $R(\Delta)$ as defined with equation (3.13). Thus both the overall loss probability and the overall average delay can be calculated by substituting the above results into equation (3.44) and (3.46), respectively.

The simputation of a FIFO queue has a similar structure and all performance measures are obtained in analogous way.

Initially I analyzed an LDOLL queue with $Q = 50$ and $\Delta = 25$. In order to compare the cell and the burst level performance, I "simputated" 10 minutes of real time. The result for both LDOLL and FIFO disciplines, are shown in figure 3.14.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure314.png}
\caption{BURST level performance improvements of the LDOLL over the FIFO queue for $Q = 50$ and measured traffic as input.}
\end{figure}

The LDOLL results are rather disappointing since the performance improvements of the LDOLL mechanism virtually disappear compared to figure 3.4. This is especially true for

\textsuperscript{14}This probability can be found from the histogram of the trace.
the LL loss probability which improves for less than an order of magnitude for the whole range of output link utilization, \( \rho \in \{0.4, 1.0\} \), with respect to the FIFO performance. Similar conclusions can be drawn from the delay graph.

Because of the disappointing results, we performed another set of experiments, where the size of the LDOLL queue was varied (increased), while the rest of the parameters remained the same. The results for both LL and LD class, are shown on figures 3.15 and 3.16, respectively.

![Figure 3.15](image1.png) ![Figure 3.16](image2.png)

Figure 3.15: Average delay and loss probability of the LL cells in an LDOLL queue for varying buffer size and measured traffic as input.

Figure 3.16: Average delay and loss probability of the LD cells in an LDOLL queue for varying buffer size and measured traffic as input.

The graphs clearly indicate that increasing the buffer size greatly improves the performance of the LDOLL queue. For instance for \( Q = 250 \), the LL cells experience almost negligible loss up to a utilization of 70\% , while the average delay of the LD cells is only twice or three times the LD delay in a buffer of size \( Q = 50 \).

Note that typical BURST level congestion for the LL class occurs by average utilization of almost 80\%, whereas the BURST level congestion for the LD cells starts when only 65\% of the link capacity is utilized. Note also that these are relatively high utilization levels, considering that half of the traffic load is bursty data traffic and all traces are self-similar.
The next set of graphs depicts a performance comparison between an LDOLL and a FIFO queue, for $Q = 250$ (compare also figure 3.17 with figure 3.14). It clearly shows the performance improvements of the LDOLL queuing for larger buffers. The figure indicates that for high utilization levels ($\rho \geq 0.8$) corresponding to the BURST level congestion region, the LL loss performance improves for two orders of magnitude in respect to both FIFO and LD loss performance. Note that for $\rho = 0.7$ the improvement is almost five orders of magnitude. The reason for such a sharp drop of the LL cell loss probability is that for the LL cells the link loading is virtually lower because they have storage priority over the LD cells. Thus the LL burst level congestion starts later, i.e. at higher utilization levels. Similar conclusions can also be drawn from the average delay graph.

In summary, the LDOLL performance improvements are in general lower than the one obtained from the CELL level analysis, but still significant\footnote{In the case considered here, the LDOLL queue could potentially provide a 15\% higher utilization.}. This is of course not surprising because of the high variability of the BURST level traffic\footnote{As discussed in the previous chapter, the employed traces exhibit self-similar properties.}.

### 3.2.3 The impact of Self-Similarity on the buffer performance

Given that self-similarity is a ubiquitous property of the measured network traffic which is not captured with the traditional traffic models, there has been an increasing concern about its impact on the performance analysis and the quality of the traffic engineering guidelines based on the analysis. This is further emphasized in the light of the initial results, indicating that the performance of queuing models with long-range dependent (LRD) input processes is much worse than estimated by the traditional short-range dependent (SRD) models (see [Nor94],[ENW96]). Thus there is a genuine need for an extensive traffic engineering analysis of the long-range dependent network traffic, including traffic generation, stochastic modeling, and performance analysis.

This subsection focuses on the later two topics. I analyze the quality of both traditional (SRD) and LRD traffic models described in chapter 2, and compare their performance with the performance of the actual network traffic. Table 3.1 lists all traffic models that
3.2 Burst level analysis

<table>
<thead>
<tr>
<th></th>
<th>LD traffic</th>
<th>LL traffic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measured traffic</td>
<td>VBR video trace</td>
<td>WWW traces</td>
</tr>
<tr>
<td>Traditional models</td>
<td>AR model</td>
<td>ON-OFF sources</td>
</tr>
<tr>
<td>Self-similar models</td>
<td>fARIMA model</td>
<td>Pareto rate model</td>
</tr>
</tbody>
</table>

Table 3.1: Categorization of the traffic models used in the analysis, according to the traffic class.

are used in this subsection and specifies the traffic class they model.

Both LDOLL and FIFO queuing are considered and the model settings are the same as in the previous subsection, namely the queue has three LL and one LD input link, and a single output link. I will discuss only the cell loss probability because it is more appropriate for this type of analysis. Note however that the similar conclusions can be derived from the average cell delay results.

The first experiment considers an LDOLL queue with $Q = 50$. The graphs on figure 3.18, depict the LD, LL and FIFO cell loss probability for both traditional traffic models (a) and self-similar traffic models (b). The results should be compared with the loss graph (left) of figure 3.14, which depicts the loss performance for the empirical measured traces. The graphs show that the traditional models slightly underestimate the actual cell loss probability, whereas the self-similar models are overly pessimistic. In other words self-similarity has much lesser impact on the performance than previously thought and the traditional models provide much closer predictions than self-similar models.

![Graphs showing cell loss probability for LDOLL queue with Q = 50](image)

(a) Traditional  
(b) Self-similar

Figure 3.18: Loss probability improvement of the LDOLL queue with $Q = 50$, for traditional (a) and self-similar (b) traffic models.

Similar results have also been reported by other authors, who focused on video traffic only (see [HL96, RE96]). A rather intuitive explanation of the apparent contradiction with the initial studies, is that early results of the ill-effects of self-similarity on the performance, have been based on the infinite buffer assumption. In finite buffers however,
whenever the buffer fills up and forces cell loss, it also "looses" memory of the overflowing traffic. Thus the buffer size imposes a natural limit on the time span over which the traffic correlations can influence the buffer performance. This hypothesis was later confirmed by other authors, [RE96, GB96, RH96].

To further validate the last results, we performed another set of experiments in the spirit of [ENW96]. Namely, we analyze the performance of a FIFO queue for varying load and buffer size ranging from 10 to 1400 cells, for all three types of traffic models. The results are depicted on figure 3.19. Figure 3.19(a) shows the performance of the actual traffic. As long as the utilization is below 60%, the loss probability decreases very fast with increase of the buffer size, which is typical for the cell level congestion regime. Then there is a very narrow transition interval around 70% utilization, after which the burst level congestion takes over, where further increase of the buffer size has almost no effect on the cell loss probability.

We are actually interested in how do figures 3.19(b) and 3.19(c) compare to figure 3.19(a). From figure 3.19(b) we see that the performance of the traditional models exhibits that same kind of behavior as the actual traffic, although slightly too optimistic, especially in the transition regime. The self-similar models on the other hand (see figure 3.19(c)), grossly overestimate the actual cell loss probability. The explanation for such a behavior can be found in chapter 2. There it was shown that the self-similar models produce accurate representation of the long-range correlations of the actual traffic and a very poor representation of the short-term correlations. The opposite is true for the traditional models. Thus the short-term correlations have a determinative effect on the burst level performance of the finite buffers, which agrees with my previous conclusion. By extension, the applicability of the engineering guidelines based on traditional models remains strong, although an additional validation effort is advisable. The results also illustrate the danger of using theoretical traffic models that are not validated with realistic traffic measurements, for performance evaluation of telecommunication systems.

There is one more point that needs attention. Namely, according to the hypothesis presented above, the impact of the self-similarity should become more significant with the increase of the buffer size. Looking at the current trends in the software development, where applications demand more and more bandwidth as the capacity of the network increase, and the continuously dropping memory prices, it is very likely that in the future the network nodes will have very large buffers. Thus the impact of the self-similarity in a presence of very large buffers is a relevant subject that needs further investigation.

### 3.3 Conclusion

This chapter is concerned with traffic control mechanisms that ensure efficient utilization of network resources while satisfying the QoS requirements of all users. In particular, it focuses on preventive control mechanisms that guarantee statistical QoS requirements.

The chapter analyzes the LDOLL queue, a CELL level priority mechanism that mixing the LD and the LL traffic in such a way that improves the performance of both traffic classes.
3.3 Conclusion

Figure 3.19: Cell loss probability in a FIFO buffer with varying size and utilization as a parameter.

It examines both the CELL level and the BURST level performance of the LDOLL queue. The CELL level analysis considers two cases, an LDOLL queue with a single outlet and an LDOLL queue with multiple outlets sharing the same buffer resources.

The retrieval policy that minimizes both the LL cell loss probability and the LD cell delay/variance in the single outlet LDOLL queue, is called the threshold policy. The single outlet LDOLL queue, governed by the threshold policy, attains very significant performance gains for both traffic types, with respect to standard FIFO queuing. The highest gains are achieved when the LD and LL load is balanced, whereas the tradeoff between the LD and LL performance is controlled by varying the threshold $\Delta$ of the threshold policy.

The retrieval policies that optimize the performance of multiple-outlet LDOLL queue have very complex structure. On the other hand, the performance of the myopic threshold policies is very close to the performance of the optimal policies. Furthermore, because of their simple structure, they are very attractive from both generalization and implemen-
ation point of view. Thus for any practical purposes the myopic threshold policies are considered the best solution for the multiple-outlet LDOLL queue. The analysis shows that the multiple-outlet LDOLL queue, governed by a myopic threshold policy, accomplishes superior performance in comparison to both shared FIFO and separate LDOLL queuing mechanisms. Similarly as in the single-outlet case, the maximal performance gains are achieved in a balanced environment.

The BURST level analysis is concerned with two main issues, the BURST level performance of the LDOLL queue and the impact of the self-similarity on the ATM buffer performance. The analysis shows that the performance improvements of the LDOLL queue at the BURST level are generally lower than at the cell level, mainly because of the high variability of the BURST level traffic. The performance improvements however, are still significant. In the case considered here, the link utilization can be increased by almost 15% compared to standard FIFO queuing. Furthermore the improvements become more pronounced for larger buffer sizes.

The main conclusion of the analysis of the performance impact of the self-similarity, is that the self-similar property of the network traffic does not have a determinative effect on the buffer performance, not for medium sized buffers at least. This is because through the occasional cell loss finite buffers also "loose" memory of the lost traffic, which limits the time span over which the traffic correlations can impact the buffer performance. Consequently traditional modeling techniques are still viable, although their applicability should be investigated on a per case basis.
Chapter 4

ATM network design

The previous chapter considered cell level mechanisms for improving the efficiency of the ATM network. In particular, it analyzed the CELL and BURST level performance of the LDOLL queuing discipline. This chapter however focuses on the impact of the performance parameters on the ATM network topology and link dimensioning. In other words, it focuses on the design of ATM networks.

As already discussed in chapter 1, circuit-switching design techniques are preferred over packet-switching design techniques because ATM is connection-oriented. The design of multi-rate circuit switching networks is a mature discipline and there exist a number of reasonably efficient design algorithms in this area [Kel86, Kel88, Kel90, Gir90]. The design of ATM networks however, differs considerably from the design of multi-rate circuit switching networks in several important aspects that significantly increase the complexity of the design process:

(i) the need for a multi-level traffic description,

(ii) the complex relationship between the offered traffic and the amount of bandwidth needed to satisfy the QoS requirements of various services,

(iii) the high variability of both the traffic load and the QoS requirements of the various services,

(iv) the high modularity of the network links;

(v) the multi-layer structure of the ATM network (VC/VP and PHY layer), etc.

The first aspect, the multi-level traffic description, was already discussed in chapters 1 and 2. In chapter 2 was shown that the ATM traffic exhibits different behavior depending on the time scale from which it is observed. Hence the need for a level specific traffic modeling in ATM networks, in contrast to the single-level traffic description, which is used in the circuit-switched network design methods.

The second aspect has to do with the minimal amount of network resources, required to support the services with acceptable performance quality. In ATM networks, analogous to the traffic behavior, there are several levels of performance, CONNECTION level, BURST level and CELL level performance (for examples see chapter 3). Furthermore the
performance of various levels are interrelated, such that decisions at the higher levels have significant impact on the performance at the lower levels. For instance increasing the blocking at the connection level can significantly lower the cell loss probability at the burst and cell level.

The third aspect follows from the ATM's ability to integrate a wide range of services with very different QoS and bandwidth requirements. Namely, ATM networks are expected to support services that require from a few kbit/s up to a several tens of Mbit/s, and QoS ranging from best-effort to a very low cell loss probabilities and tightly controlled delays.

The fourth aspect is concerned with the difficulties that result from the very large size of the network links compared to the bandwidth requirements of some services. For instance, consider a typical link size of 150 Mbit/s (OC3), which carries traffic of various services with the smallest rate being 16 kbit/s. In terms of the circuit switched design model, the sample link is viewed as having 9375 circuits. Furthermore the number of services supported by the ATM network is also expected to be high. Since the computation effort of the most of the circuit-switched design algorithms, increases polynomially in both the size of the network links, expressed in number of circuits, and the number of supported services, it is clear that for typical ATM values the operation of the algorithms may easily become prohibitively expensive.

The fifth aspect has to do with multiple layers that form the structure of an ATM network. As already discussed in chapter 1, the ATM network is composed of three layers: virtual connection (VC), virtual path (VP) and the physical (PHY) layer. The PHY layer comprises the facility network, i.e. the SDH or SONET trunks and cross-connect switches. On top of the physical network, a number of VP overlay networks coexist, deterministically sharing the same physical resources. In turn many VCs are statistically multiplexed within a single VP overlay network. The design complexity stems from the fact that network topology at each layer need not to be the same and the link dimensioning at all three layers is mutually dependent.

From the discussion it is clear that ATM network design requires extension of the current circuit-switched design methods as well as development of new design algorithms. This chapter describes an ATM design method that considers all of the aspects discussed above, and in particular, the tradeoffs that result from (v). First, the next section introduces two important concepts that are particular to the ATM networks and discusses the design tradeoffs that follow from their implementation. Then Section 4.2 introduces the network model and formulates the design problem. Section 4.3 then discusses the previous work on the subject and presents a detailed description of the design framework. Next, section 4.4 gives a short description of the software tool that implements the design framework, section 4.5 presents some preliminary results and finally section 4.6 concludes the chapter by summarizing the research results.

4.1 Introduction

In the last decade, extensive research of aspects (i) and (ii), has resulted in introduction of the concept of effective bandwidth. The idea is that each traffic source is assigned a rate at the connection level, which lies between its mean and peak rate, that ensures
that its QoS requirements will be satisfied at the BURST and CELL level. From a network
design point of view, the concept is very natural. In essence it hides the details of the lower
levels, by relating the BURST and the CELL level traffic and performance parameters to the
CONNECTION level parameters. This separation of levels, is essential for the tractability
of the network design problems.

There exist many approximations for calculation of the effective bandwidth depending
on the type of QoS requirements of the service (deterministic or statistical guarantees) and
its traffic description [CLW96, GAN91, EM93, KWC93, GH91]. However, two properties are
common to most approximations, namely:

- the independence property: the effective bandwidth of a traffic source is independent
of the characteristics and requirements of other sources with which it is multiplexed,

- the additivity property: the sum of the effective bandwidths of two independent
sources is equal to the effective bandwidth of their superposition.

Thus the complexity of computing the effective bandwidth depends only on the source
dimension, not on the system dimension, which in turn greatly simplifies the connection
admission control (CAC). The basic idea is to assign each source an effective bandwidth
requirement, and then consider a newly arrived call admissible if the sum of the effective
bandwidths of the newly arrived call and the ongoing calls, is less than the available
bandwidth on any feasible network route. Thus the effective bandwidth concept is very
suitable when solving higher dimension design problems.

Most of the research efforts have focused on determining the effective bandwidth
for various traffic models and FIFO buffers with constant service rate. Relatively lit-
tle is available on effective bandwidths when priority queuing disciplines are used (see
[KmC95, IEE95]). The major obstacle in the analysis of the effective bandwidths in
multi-priority queues lies in the strong dependence of the performance of one class on
the traffic parameters of the other classes. This is also true for the LDOLL queue, see
for example sections 3.1.2 and 3.1.3. Since the CONNECTION level analysis of the LDOLL
queue is beyond the scope of this thesis, developing an effective bandwidth approximation
in case of LDOLL queuing is left for future research activities.

The virtual path (VP) layer plays a very central role in the ATM network. With its
introduction in the ATM network structure, a new degree of freedom has been added to the
network design and management functions. Namely, a number of virtual subnetworks can
coexist on the VP layer, sharing the same physical transmission and switching resources.
The subnetwork types range from a simple subset of VPs dedicated to a single service
class to a very complex virtual private networks involving all service classes (an example
of the former type, a connectionless enterprise overlay network or a virtual LAN, will be
discussed in the following chapter).

Originally the VP concept was introduced in order to improve the operational flex-
ibility and decrease the network control and switching costs. For instance, if a certain
physical link fails, only the VPs need to be rerouted since the VCs will be automatically
rerouted too, decreasing therefore the network control costs. Another example is when the
newly arrived calls use already existing VPs to setup a connection to their destinations,
which lowers the setup costs.
However in the last couple of years, it has been gradually recognized that one of the main advantages of the VP subnetwork concept, is improvement of network resource utilization. Namely, in the absence of sophisticated priority queuing mechanisms at the CELL level, integrating all services in one network would imply a highly variable distribution of the QoS requirements. As a result, every connection would be provided with effective bandwidth corresponding to the most stringent requirement, leading to a huge overdimensioning of the network (see e.g. [RMV96]). Thus it seems more efficient to support different services by grouping them into several clusters of similar QoS requirements, and then configuring a separate VP subnetwork for each cluster. This effectively limits the degree of integration to only partial, rather than complete, sharing of physical transmission and switching resources, so as to provide each service with quality not much higher than the one required.

Moreover, it can be reasonably argued that control operations, such as call admission, routing and policing, can be strongly simplified in a network with only quasi-homogeneous traffic. For example, it is intuitively clear that the four service classes discussed in section 2.1, are handled more efficiently when separated in three different VP subnetworks, rather than when they are multiplexed on a complete sharing basis. Note also that although the virtual subnetworks share the physical resources deterministically, statistical multiplexing, priority queuing and other mechanisms can still be applied within a single subnetwork.

When considering designing an ATM network, a question that follows naturally from the previous discussion, is what is the most appropriate way of clustering the network, i.e. which VP subnetwork configurations yields the most efficient ATM network? For instance, following again the traffic classification of section 2.1, what yields the biggest cost savings (or alternatively the highest throughput), configuring a separate VP subnetwork for each traffic class, or one separate network for the LX class and another subnetwork that integrates all other classes, or maybe a separate subnetworks for the LX and HX classes and a single subnetwork for the LD and LL classes that uses the LDOLL queuing mechanism at the CELL level?

The overall efficiency of the ATM network is also dependent on how the network resources are shared between the various subnets. Since the VP subnetworks share the same physical transmission and switching capacities, there is a tradeoff between their quality. For example, the grade of service (GoS) parameters\footnote{The GoS parameters characterize the connection level performance, whereas the QoS parameters characterize the burst and cell level performance.}, like the call blocking probability, in one of the subnetworks can only be improved by degrading the GoS parameters in other subnetworks.

Moreover, the overall efficiency is also affected by the distribution of traffic among various VCs within a single VP subnetwork. The design problem is further complicated by the fact that the last aspect is closely related to the dimensioning of the VP links (see section 4.3.4).

It is a highly non trivial task, to determine the VP subnetwork configuration along with the internal VC traffic distribution, that minimizes the network cost under the given
4.2 Problem formulation

In general ATM network design comprises several canonical problems. This chapter focuses on one of them, namely, configuring an "optimal" set of VP subnetworks on top of a given physical infrastructure, which minimizes the network cost for given traffic demands with varying QoS requirements. All of the addressed questions are at the CONNECTION and PATH levels. The analysis relies on the concept of effective bandwidth for capturing the BURST and CELL level behavior, including the related QoS issues.

The next subsection lays the ground for formulation of the network design optimization problem, by introducing a network model that captures all of the relevant design issues. The optimization problem is defined in the subsequent subsection.

4.2.1 Network model

The model assumes a fixed physical infrastructure consisting of \( N \) nodes and \( K \) physical links connecting them. A number of VP subnetworks can be configured on top of this physical infrastructure, by forming logical links at the VP layer. A logical link, also called a VP link, may traverse several physical links. Each subnetwork consists of a subset of VP links forming a virtual topology. For practical purposes, any feasible virtual topology must be connected in a graph theoretic sense. In principle the topology of a VP subnetwork may differ from the physical network, in both number of links and number of nodes. The number of VP subnetworks and their topology, as well as the distribution of services among various subnetworks, are all variables of the model.

Let \( V \) denote the number of VP subnetworks configured on top of the physical network, and \( P = \{ P_1, \ldots, P_V \} \) their topologies. Every component of \( P \), \( P_v \), is in fact a two dimensional \( (N \times N) \) matrix, whose elements are defined as:

\[
p_{vij} = \begin{cases} 
1 & \text{if there is a VP link between nodes } i \text{ and } j \text{ in subnetwork } v \\
0 & \text{otherwise.}
\end{cases}
\] (4.1)

Note that this implies that each \( P_v \) is a matrix of binary decision variables. Furthermore let \( L \) denote the total number of VP links in all subnetworks, and \( C_l, 1 \leq l \leq L \), the capacity of the VP \( l \). The VP links are unidirectional, and links with opposite directions may have different capacities. The mapping of the VP links of all subnetworks on the physical infrastructure, is specified with a \( K \times L \) binary matrix \( M \), whose elements are defined as:

\[
m_{kl} = \begin{cases} 
1 & \text{if VP } l \text{ traverses physical link } k, \\
0 & \text{otherwise.}
\end{cases}
\] (4.2)
Since the physical network resources are shared deterministically between the configured VP subnetworks, the sum of capacities of all VPs traversing any physical link $k$, must never exceed its physical capacity $C_k^{\text{phy}}$. Mathematically, this constraint is expressed with the following inequality:

$$MC \leq C^{\text{phy}}$$  \hspace{1cm} (4.3)

where $C = \{C_1, \ldots, C_L\}$, and $C^{\text{phy}} = \{C_1^{\text{phy}}, \ldots, C_K^{\text{phy}}\}$. Finally, the script letters, $\mathcal{N}$, $\mathcal{K}$, $\mathcal{V}$ and $\mathcal{L}$, denote the sets of nodes, physical links, VP subnetworks and VP links, respectively.

It is assumed that $S$ traffic types are carried by the network. The role of the traffic types is primarily to deal with various QoS requirements requested by different services, but traffic types can also be formed with respect to different holding times or priorities.

Let $\sigma$ denote an origin-destination (O-D) pair of nodes, $\sigma \in \mathcal{N} \times \mathcal{N}$, and $(v, s, \sigma)$ a stream of calls of traffic type $s$ from the O-D pair $\sigma$, carried by the subnetwork $v$. Traffic streams of every service are modeled at both CONNECTION and BURST level. The call arrival process at the CONNECTION level, is modeled as a Poisson process with mean arrival rate $\Lambda_{v\sigma}$, whereas the call holding times are assumed to be generally distributed with mean $1/\mu_{v\sigma}$ and independent of earlier arrival and holding times. At the BURST level, we use the ON-OFF model described in section 2.3 with three traffic parameters: peak rate, mean OFF time and mean ON time, denoted by $R_{\text{peak}}^{\text{v\sigma}}$, $1/\theta_{v\sigma}$ and $1/\eta_{v\sigma}$, respectively. The main purpose of the BURST level modeling is to allow for calculation of the effective bandwidth of the $(v, s, \sigma)$ traffic stream. Thus the external traffic is characterized by five four-dimensional matrices $(V \times S \times N \times N)$: $\{\Lambda_{v\sigma}\}$, $\{\mu_{v\sigma}\}$, $\{R_{\text{peak}}^{\text{v\sigma}}\}$, $\{\theta_{v\sigma}\}$, and $\{\eta_{v\sigma}\}$.

The routes followed by the $(v, s, \sigma)$ traffic streams, play an important role in the network model. A route $r$ is basically a subset of VP links of a certain VP subnetwork, that forms a virtual connection from the origin to the destination node. By convention, each route carries traffic of a single service type $s$. In other words, if several services are to be carried between the same $\sigma$ pair, they are represented by parallel routes. The length of a route in VP hops, is limited by the delay requirement of the corresponding service. The set of admissible routes for traffic stream $(v, s, \sigma)$ is denoted by $\mathcal{R}(v, s, \sigma)$. The model does not employ alternative routing, hence throughout the chapter, the routing sets $\{\mathcal{R}(v, s, \sigma)\}$ are assumed to be fixed and independent of the state of the network.

Another important tradeoff at the VC layer is how to distribute the traffic load among the routes carrying the same traffic type between origin-destination (O-D) pair, $\sigma$. In principle, a single $(v, s, \sigma)$ traffic stream is spread over multiple routes between the origin and the destination node. This is modeled with a Bernoulli process. The outcome of an independent Bernoulli trail determines if an arriving call of stream $(v, s, \sigma)$ is offered to the network, which is a form of an admission control, and, if it is, to which route $r \in \mathcal{R}(v, s, \sigma)$ it is offered to. Furthermore, if an arriving call in the stream $(v, s, \sigma)$ is offered to a route $r \in \mathcal{R}(v, s, \sigma)$, it can still be blocked if there is insufficient residual bandwidth at any of the VP links along the route $r$. Therefore it always holds that

$$\sum_{r \in \mathcal{R}(v, s, \sigma)} \lambda_{v\sigma r} \leq \Lambda_{v\sigma},$$  \hspace{1cm} (4.4)
where $\lambda_{usr}$ denote the mean Poisson call arrival rate of service $s$, on route $r \in \mathcal{R}(v, s, \sigma)$. The corresponding traffic intensity on route $r$ is defined as $\rho_{usr} = \lambda_{usr} / \mu_{usr}$. Then equation (4.4) becomes:

$$\sum_{r \in \mathcal{R}(v, s, \sigma)} \rho_{usr} \leq \bar{\rho}_{usr},$$

(4.5)

where $\bar{\rho}_{usr} = \Lambda_{usr} / \mu_{usr}$. The Bernoulli success probability of the admission control function, can then be found by:

$$p_{usr} = \frac{1}{\Lambda_{usr}} \sum_{r \in \mathcal{R}(v, s, \sigma)} \lambda_{usr},$$

(4.6)

This basically means that the external calls of stream $(v, s, \sigma)$, are offered to the subnetwork $v$ with the probability $p_{usr}$. Similarly the probability that a call of traffic type $s$, is offered to route $r \in \mathcal{R}(v, s, \sigma)$, is equal to $\lambda_{usr} / \Lambda_{usr}$. Both probabilities are determined by the design process.

The last parameter of the network model that needs to be specified is the amount of bandwidth at the connection level, that a route $r$ requires on link $l$, i.e. the effective bandwidth of the traffic type $s$ on a certain VP link. Let $d_{est}$ denote the effective bandwidth that traffic type $s$ requires on link $l$ in subnetwork $v$. The notation implicitly assumes that all routes carrying traffic type $s$ require the same amount of bandwidth on link $l$. Since the effective bandwidth is closely associated with the traffic type this is not a restriction. On the other hand the effective bandwidth requirement of calls on a given route, may vary along the links of the route. This is necessary since the effective bandwidth of services requesting deterministic QoS guarantees, is also dependent on the capacity of the involved links, as discussed later in the chapter (see section 4.5).

To recapitulate, the important elements of the model are the fixed set of nodes and physical links, a set of VP subnetworks carrying various mixes of services mapped on the physical network, and a set of admissible routes obtained by a state-independent routing algorithm. Additionally, the offered traffic load of $(v, s, \sigma)$ stream is subject to admission control, after which the offered load of service $s$ on route $r$ in subnet $v$, is $\rho_{usr}, r \in \mathcal{R}(v, s, \sigma)$.

### 4.2.2 Optimization problem formulation

The VP subnetwork design problem can be formulated as a distributed network optimization problem. Using the network model defined in the previous subsection, the problem is defined as follows:
Given:
\[ \mathcal{N}, \mathcal{C}^{\text{phy}}, \mathcal{K}, \mathcal{V}, \mathcal{RSS}, \mathbf{D}_{\text{max}}, \{\Lambda_{\text{usa}}\}, \{\mu_{\text{usa}}\}, \{P_{\text{usa}}^{\text{peak}}\}, \{\theta_{\text{usa}}\}, \{\eta_{\text{usa}}\} \]

Find:
\[ M^*, \mathbf{P}^*, \mathbf{C}^*, \{\rho_{\text{usa}}^*\} \quad \forall r \in \mathcal{R}(v, s, \sigma), \]
\[ \forall v \in \mathcal{V}, \forall s \in \mathcal{S}, \forall \sigma \in \mathcal{N} \times \mathcal{N} \]

Minimizing:
\[ \alpha Z_B(M, \mathbf{P}, \mathbf{C}, \rho_{\text{usa}}) + \beta Z_c(M, \mathbf{P}, \mathbf{C}, \rho_{\text{usa}}) \quad (4.7) \]

Subject to:
\[ \sum_{r \in \mathcal{R}(v, s, \sigma)} \rho_{\text{usa}} \leq \overline{\rho}_{\text{usa}} \quad \forall v \in \mathcal{V}, \forall s \in \mathcal{S}, \forall \sigma \in \mathcal{N} \times \mathcal{N} \]
\[ MC \leq \mathcal{C}^{\text{phy}} \quad (4.8) \]
\[ f(r) \leq 1_s^\top \cdot \mathbf{D}_{\text{max}} \quad \forall r \in \mathcal{R}(v, s, \sigma), \]
\[ \forall v \in \mathcal{V}, \forall s \in \mathcal{S}, \forall \sigma \in \mathcal{N} \times \mathcal{N} \quad (4.9) \]

where RSS stands for the Resource Sharing Strategy and \( \mathbf{D}_{\text{max}} \) denotes the service delay requirement vector. The RSS determines how the services will be distributed among the various VP subnetworks, i.e., it determines which subnetwork supports which service. The \( \mathbf{D}_{\text{max}} \) determines the maximal number of VP and VC switching nodes on route \( r \), for each service, based on its delay requirement.

The objective function (4.7), is a linear combination of two terms, \( Z_B(M, \mathbf{P}, \mathbf{C}, \rho_{\text{usa}}) \), and \( Z_c(M, \mathbf{P}, \mathbf{C}, \rho_{\text{usa}}) \). The \( Z_B(M, \mathbf{P}, \mathbf{C}, \rho_{\text{usa}}) \) denotes the costs of lost revenue due to the call blocking, whereas \( Z_c(M, \mathbf{P}, \mathbf{C}, \rho_{\text{usa}}) \) denotes the costs of both VC and VP switching and VC setup. In other words the goal is to design a VP subnetwork configuration, that minimizes both the number of blocked calls of all services and the network control costs.

The call blocking cost term is defined as the difference between the offered and the carried traffic. This can be expressed as follows:
\[ Z_B = \sum_{v, s, \sigma} \sum_{r \in \mathcal{R}(v, s, \sigma)} \overline{d}_{\text{usa}} \rho_{\text{usa}} L_{\text{usa}}, \quad (4.10) \]

where \( \overline{d}_{\text{usa}} \) denotes the average effective bandwidth on all routes for the O-D pair \( \sigma \), for service \( s \), in subnetwork \( v \). The \( L_{\text{usa}} \) denotes the blocking probability of service \( s \), on route \( r \), in subnetwork \( v \). As it is shown later, \( L_{\text{usa}} \) is a very complicated non-linear function. Note also that equation (4.10) neglects the blocking as a result of the admission
4.3 The solution approach

control. This is not really a problem since the actual blocking is only marginally higher, except in heavy traffic situations.

The control cost term comprises two parts, the switching cost part and the call setup cost part. The switching part takes into account both VP and VC switching costs. Assuming a linear relationship between the cost and the average switched bandwidth [RMV96], the switching costs can be expressed as:

\[ Z_{\text{switch}} = \sum_i [\zeta_{VC} + (n_i^{\text{phy}} - 1)\zeta_{VP}] \cdot a_i, \]  

(4.11)

where \( \zeta_{VC} \) and \( \zeta_{VP} \) denote the VC and VP switching cost per bandwidth unit, respectively, \( a_i \) is the average bandwidth carried on VP \( l \), and \( n_i^{\text{phy}} \) is the number of physical links traversed by VP \( l \). Note also that typically \( \zeta_{VC} \gg \zeta_{VP} \).

The call setup part is assumed proportional to the average call arrival rate and to the number of VP links tested during the call setup phase, thus:

\[ Z_{\text{setup}} = \sum_{v,s,\sigma} \zeta_s \sum_{r \in R(v,s,\sigma)} \lambda_{\text{surf}} Y_{\text{surf}} \]  

(4.12)

where the \( \zeta_s \) is the unit setup cost per call per VP involved in the routing procedure and \( Y_{\text{surf}} \) is the average number of VPs involved in the call setup phase.

The first two sets of constraints were already discussed in the previous subsection, see equations (4.3) and (4.5). The last set of constraints (4.9), ensures that the delay requirements of all services are satisfied. This is done by limiting the number of switching buffers that an admissible route can traverse. The function \( f(r) \) returns the number of VP and VC switching nodes on route \( r \). The \( 1_s \) is just a column vector with all zeros and a single 1 for the service \( s \). Thus each equation from the (4.9) set, determines the maximal number of VC and VP switching nodes in the feasible route set, of service \( s \).

The design problem is clearly very difficult. It contains both discrete and continuous variables and a non linear objective function constrained over a convex set. Because of the complexity, the problem can not be solved exactly. Therefore the next section presents a heuristic solution procedure that tries to find a solution by iteratively solving a sequence of relevant subproblems.

4.3 The solution approach

The VP subnetwork design problem, which is described in the previous section, is NP-hard because several of its subproblems are NP-hard. It can be partitioned into several, mutually dependent subproblems:

1. Determine the optimal virtual topologies of all VP subnetworks,
2. Route the VP links of all subnetworks over the physical network,
3. Route the VC traffic over the VP links and determine the optimal size of the VP links in the whole network.
Subproblem 1 addresses how to most efficiently transmit the network traffic, subproblem 2 addresses how to properly utilize the limited amount of available network resources, and subproblem 3 addresses how to simultaneously optimize the VC routing and the capacity of the VP links, also known as the capacity and flow assignment (CFA) problem in packet-switched networks [GK77, Ker93].

Most of the previous research has focused on the development of efficient algorithms for a single sub-problem. Faragó et.al. [FBA+95] consider subproblem 3 and propose a solution procedure based on standard convex programming methods. Mitra et.al. [MMR96] developed a computationally very efficient gradient algorithm for solving the same sub-problem, independently of [FBA+95]. However their problem formulation considers only a single VP subnetwork. Chlamtac et.al. [CFZ94] consider subproblem 2 and provide a very general algorithm for routing of the VPs that minimize the maximum load on the physical links.

The general design problem has been also studied before. For instance, Yan and Beshai [YB95a, YB95b], consider a broadband ATM network design procedure which assumes that all services are multiplexed onto the same VP subnetwork (CS multiplexing strategy). Chapter 11 in [RMV96], discusses several design algorithms that incorporate all of the three subproblems, described previously.

Our approach differs from the past research in that the formulation is more general, in the sense that it encompasses all of the above subproblems and multiplexing strategies and it also accommodates constraints on the physical link capacities and permissible average cell delay, which have not been considered earlier. This allows for investigation of the tradeoff between the qualities of different subnetworks. Furthermore it also allows for preliminary investigation of the impact of various strategies for multiplexing traffic types that differ in both delay and loss QoS requirements, like the LDDOLL queue.

As already mentioned, the problem at hand is a very complex, non linear, mixed integer program, whose objective function is neither convex nor concave. Therefore the proposed solution involves a heuristic procedure, that yields a locally optimal network configuration. The next subsections describes the global design framework and each of the relevant optimization steps.

### 4.3.1 The design framework

The global design framework is a heuristic procedure, that tries to solve the global problem by iteratively solving a sequence of simpler subproblems. It comprises five steps (see figure 4.1), namely: the Initialization step, the VP-mapping step, the Routing and dimensioning step, the Convergence control step and the Topology optimization step.

The initialization step specifies all necessary initial parameters of the optimization procedure, that do not change in the course of a single optimization run. Typical examples are the traffic model parameters, the QoS requirements, the cost parameters, the

---

2One of the algorithms discussed in Chapter 11 in [RMV96] has very similar problem formulation. Our formulation however is slightly more general because the cost function is defined as a linear combination of both performance and cost related terms, compared to the purely cost related terms considered in [RMV96].
4.3 The solution approach

Figure 4.1: The block diagram of the global design framework.

optimization algorithm parameters etc. Additionally it also determines the number of VP subnetworks, their initial topology and the distribution of the services among the subnetworks. Once the initial configuration is determined, the VP mapping step is entered.

The VP mapping step determines the routes\(^3\) of the VP links of all subnetworks in the physical network, i.e. it determines how the physical bandwidth is shared between the VP subnetworks. Note that this corresponds to solving subproblem 2. The difficulty stems from the fact that the optimal VP route set should be determined without knowing the loading of the VP links because the VC traffic has not been routed yet.

The Routing and dimensioning step is the most sensitive part of the procedure, because the quality of the global solution depends critically on decisions made in this step. As the name suggests it basically optimizes both the distribution of the VC traffic among various routes and the capacity of the VP links, with respect to the constraints defined in subsection 4.2.1. Note that this step corresponds to solving subproblem 3.

With the completion of the Routing and dimensioning step, the network configuration is completely specified and the network costs can be calculated. This is done in the Convergence control step. This step basically checks whether additional iterations of the global design procedure are necessary, based on certain convergence criterion. If the convergence criterion is not satisfied the procedure begins a new iteration by entering the Topology optimization step. The Topology optimization step generates new subnetwork topologies following a certain optimization algorithm in order to find the one which

\(^3\)The routes of the VPs in the physical network should not be confused with the VC routes, which are a subset of VP links forming a virtual circuit.
minimizes the objective function of the global problem. Note that the last two steps correspond to solving subproblem 1.

Although the separation of the global problem into several subproblems is heuristic in nature, it is also one of the strengths of the design framework from a flexibility viewpoint. For instance, suppose that the IDOLL queuing is employed in the network. Then one should make changes only in the Routing and the dimensioning block. The rest of the procedure remains virtually unchanged. Furthermore, one also has the freedom of experimenting with various algorithms for each subproblem in order to adapt to slight changes in the problem formulation or to simply test a newly developed algorithm for one of the subproblems.

In summary, procedure works as follows. For each new subnetwork topology configuration, it determines the optimal mapping of the VP links on the physical network and the optimal VC routes and VP capacities. The new topology configurations are generated according to a certain random optimization algorithm. The optimization process proceeds until the convergence criterion is satisfied. The following three subsections contain a detailed description of the solution algorithms for the Topology optimization and Convergence control, the VP mapping, and the Routing and dimensioning step, respectively.

4.3.2 Topology optimization and Convergence control

The purpose of the topology optimization step is to determine the topology of the VP subnetworks that minimizes the cost function (4.7). The problem at hand is NP-hard, hence only a fraction of the solutions can be checked within a reasonable time, rendering the enumerative algorithms useless. Furthermore the cost function is non-linear and it has many local minima that hide the global optimum, which makes the application of analytical algorithms very difficult4.

Randomized algorithms on the other hand, are independent of the structure of the cost function and the functional constraints and easily applicable to a wide range of problems. This is a result of the way these algorithms operate. Typically a randomized algorithm combines a random search with a directed search, based on some apriori knowledge about the problem structure. For our case, random algorithms are the preferred solution because of the complexity of the global design problem. Examples of such algorithms are Simulated Annealing (SA) [AvL85] and Genetic Algorithms [TB96]. Both have been employed in a variety of different application areas [JAMS89, JAMS91, VK83, Lin65, KGV83, TB96]. Currently, the design framework uses an implementation of the SA algorithm in the topology optimization step5.

Simulated Annealing is rather elegant generic optimization algorithm, based on the analogy between the annealing (cooling) of a solid, as described by the theory of quantum mechanics, and the optimization of a system with many degrees of freedom. It has been successfully applied to cases ranging from VLSI chip design [VK83], to the classical

---

4The efficiency and applicability of the analytical algorithms, is highly dependent on form of the cost function and the functional constraints. Typically complex non-linear problems can not be solved directly with an analytical algorithm.

5Other random algorithms can also be implemented easily.
4.3 The solution approach

traveling salesman problem [Lin65]. The implementation of the algorithm is typically high problem dependent. A good generic description can be found in [AvL85].

The operation of the SA algorithm is analogous to the cooling process of materials in nature. If the cooling happens gradually, the material reaches the lowest possible energy state. The algorithm repeatedly generates a random neighboring configuration from the current optimum. The new configuration is accepted as the new optimum, if it is cheaper (lower energy) than the current optimum. This corresponds to always following the "downward" path (see figure 4.2). The SA algorithm tries to avoid being stuck in a local minimum, by occasionally accepting configurations that are more expensive (higher energy) than the current optimum, with probability $\Pr\{Z_2 \mid Z_1\} \approx \exp(-(Z_2 - Z_1)/T)$, where $Z_i$ denotes the network cost according to (4.7) and $T$ is the current temperature\(^6\) (see figure 4.2). In the course of the optimization process the temperature $T$ decreases gradually, according to a certain cooling schedule. Thus the probability of accepting a more expensive configuration becomes ever lower, until at very low temperatures virtually no "upward" moves are allowed. The idea is to "wander" through different VP configurations at high temperatures, and hopefully reach the configuration with lowest possible cost in the end.

![Figure 4.2: An illustration of a two-dimensional sample path of the Simulated Annealing optimization process.](image)

Although the implementation of the algorithm is highly problem dependent, it always includes the following four elements:

1) a system configuration description,

2) a generator of random configuration changes or neighborhoods,

3) an objective function and

\(^6\)The $T$ is a dimensionless control factor of the algorithm, analogous to the physical temperature in the real annealing process. Hence its name.
4) a cooling schedule.

In the case considered here, the system configuration is determined by the location of the VP links in all subnetworks, i.e. the $P$ variable. Configuration changes are done in two ways: by changing the topology of all subnetworks or of just one subnetwork. At high temperatures, the topology of all subnetworks are changed in every trial, whereas at low temperatures, the topology of only one randomly chosen subnetwork is altered. In both cases a new configuration is generated by exchanging the VP links of two randomly chosen network nodes. This means that if node $i$ is connected to nodes $p$ and $q$, and node $j$ is connected to nodes $x$ and $y$, then after an exchange operation, node $i$ will be connected to nodes $x$ and $y$, whereas node $j$ will be connected to nodes $p$ and $q$. Additionally, if $i$ and $j$ are adjacent nodes, the VP link between them will be deleted, and vice versa, if they are not adjacent, a VP link will be created between them. In this way the topology of the subnetwork can vary between a full mesh and a minimum spanning tree.

The objective function in our case, corresponds to the network cost and it is the same as the cost function of the global problem, (4.7). After a configuration topology change, one must also specify the $M$, $C$ and $\{\rho_{urr}\}$, in order to calculate the network cost. Thus the VP mapping and the Routing and dimensioning steps must be performed, before calculating the network cost in the Convergence control block.

The Convergence control block implements the cooling schedule. The cooling schedule is based on the algorithm proposed in [JAMS89, JAMS91]. The initial temperature is chosen as a certain percentage, INITTEMP, of the cost of the initial solution. The temperature is kept constant for $L$ trials. $L$ is defined as $L = H \cdot SIZEFACTOR$, where SIZEFACTOR is an input parameter and $H$ is the expected neighborhood size. A neighborhood is defined as a set of configurations that can be reached from the current configuration with a single change. From the exchange rule, it can be easily derived that the neighborhood size per subnetwork is $h_i = N_i(N_i - 1)/2$, where $N_i$ denotes the number of nodes in subnetwork $i$. Thus when changes are made in all subnetworks simultaneously, the neighborhood size is just the product of all $h_i$s, and when changes are made in only one subnetwork, the neighborhood size corresponds to the respective $h_i$.

After $L$ configuration changes, the temperature is decreased with a decay factor $t_{\text{decay}}$ according to $T \leftarrow T \cdot t_{\text{decay}}$. If the acceptance probability exceeds the CUTOFF value, the temperature is decreased without completing all $L$ changes. If $L$ changes are completed while the percentage of accepted moves is less than MINPERCENT, a counter is incremented. The counter is reset each time a solution is found that is better than the previous best solution. When the counter reaches the value FROZEN, the annealing stops and current best solution becomes the global solution.

Thus the algorithm generates random subnetwork topology configurations and repeatedly iterates through the cycle given in figure 4.1, until the FROZEN condition is satisfied. Every iteration includes also the VP mapping and the Routing and dimensioning steps. In the following two subsections both steps are discussed in some more details.

### 4.3.3 VP mapping

The purpose of the VP mapping step is to find feasible routes of the VP links of all subnetworks in the physical network, that minimize a selected objective function. In
4.3 The solution approach

other words, this step determines the optimal value of M, M*. The objective function can express different objectives, e.g. increasing network robustness by minimizing the maximum link loading or maximizing network revenue by maximizing the carried traffic, etc.

Since the problem in general is NP-complete, one is interested in approximating solution algorithms with feasible computation complexity. In [RMV96], the authors discuss several VP mapping algorithms with different objectives and varying complexity. For the problem at hand, we have used a routing algorithm, proposed in [CFZ94], that minimizes the maximum physical link overload ratio, in polynomial time.

The choice is based on the following three properties of the algorithm:

(i) it is computationally as complex as a standard shortest-path routing algorithm,

(ii) in spite of its apparent simplicity, it always yields solutions deviating only for a small factor from the best possible solution, and

(iii) the optimality of the solution is guaranteed even when the VP traffic demands are not explicitly specified.

The last property is of significant importance, because at this stage of the design framework, the loading of the VP links has not been determined yet\(^7\). Therefore it is very useful to have an algorithm that guarantees optimal VP routing, regardless of the VP traffic distribution.

The algorithm is called Random Path (see [CFZ94]), and it can be described as follows:

**Step 1:** Set \( i = 1 \).

**Step 2:** Draw a random permutation \( \sigma(1), \ldots, \sigma(K) \) of the numbers \( 1, \ldots, K \).

**Step 3:** Assign the auxiliary weight \( \omega_j = 2^{L+1} + 2^{\sigma(j)} \), to each physical link \( j, 1 \leq j \leq K \).

**Step 4:** Find the minimum weight path for VP link \( i \) in the physical network.

**Step 5:** Stop if \( i = L \), otherwise increment \( i \) and go to **Step 2**.

In **Step 4**, one can use any of the well known standard shortest path algorithms [Sch87, Ker93]. The polynomial-time complexity of the algorithm, follows from the same property of the employed shortest path algorithm. It is not trivial however, to show the quality of the result. It is proven in [CFZ94], that for large networks \((K \geq 40)\), the set of VP routes provided by the algorithm yields a solution that is within a small factor of the best possible, with respect to the maximum link overload ratio\(^8\).

Furthermore it is also guaranteed that for any possible realization of the weights, the minimum weight path is necessarily a minimum-hop path. The selected route is actually randomly chosen from the set of the shortest paths. Note that considering only the shortest paths is not really a restriction, because a low hop count also minimizes the average delay of the network traffic. Similarly, considering only a single shortest path for each VP link is not seen as a restriction. The algorithm can be easily extended in direction of using multiple shortest routes for certain VPs, by adding them several times in the \( i \) list. As a result, those VPs will be automatically routed on several shortest routes.

\(^7\)This is done later in the Routing and dimensioning step.

\(^8\)More details about the algorithm can be found in [CFZ94].
### 4.3.4 Routing and Dimensioning

The purpose of the Routing and dimensioning step is to determine the optimum values of the last two remaining design variables, C and \( \{\rho_{usr}\} \). In other words, it should determine the optimal distribution of the VC traffic among the admissible VP routes in each subnetwork, and in the same time it should determine the optimal capacity of the VP links of all subnetworks, with respect to the objective function and the set of constraints. Since the control costs, \( Z_c(M,P,C,\{\rho_{usr}\}) \), are predominantly determined by the subnetwork topologies and the VP mapping, but mainly for reasons of simplicity, the objective function of this subproblem is reduced to the blocking cost term \( Z_B(M,P,C,\{\rho_{usr}\}) \). Furthermore, instead of minimizing the blocking costs, the objective function is transformed into maximizing the network revenue, i.e. maximizing the carried traffic.

Let \( e_{usr} \) be the revenue earned per carried call per unit of time, by calls of service \( s \) on route \( r \) in subnetwork \( v \), and let \( L_{usr} \) be the equilibrium loss probability of service \( s \) on route \( r \in R(v,s,\sigma) \). Then the carried traffic of service \( s \) on route \( r \) in subnetwork \( v \) is given by \( \rho_{usr}(1 - L_{usr}) \), and the long-run average network revenue, \( W \), is defined as:

\[
W = \sum_{v,s,\sigma} \sum_{r \in R(v,s,\sigma)} e_{usr} \rho_{usr}(1 - L_{usr}).
\]  

(4.13)

Therefore the reduced optimization problem can now be formulated as follows:

**Given:**

\( \mathcal{N}, C^{phy}, K, V, RSS, D_{max}, M, P \)
\( \{\Lambda_{usr}\}, \{\mu_{usr}\}, \{\rho_{peak}\}, \{\theta_{usr}\}, \{\eta_{usr}\} \)

**Find:**

\( C^*, \{\rho_{usr}^*\} \)

\( \forall r \in R(v,s,\sigma), \quad \forall v \in V, \forall s \in S, \forall \sigma \in \mathcal{N} \times \mathcal{N} \)

**Maximizing:**

\[
W = \sum_{v,s,\sigma} \sum_{r \in R(v,s,\sigma)} e_{usr} \rho_{usr}(1 - L_{usr})
\]  

(4.14)

**Subject to:**

\[
\sum_{r \in R(v,s,\sigma)} \rho_{usr} \leq \bar{\rho}_{usr} \quad \forall v \in V, \forall s \in S, \forall \sigma \in \mathcal{N} \times \mathcal{N}
\]

MC \leq C^{phy}

\[
f(r) \leq 1_s^\top \cdot D_{max} \quad \forall r \in R(v,s,\sigma), \quad \forall v \in V, \forall s \in S, \forall \sigma \in \mathcal{N} \times \mathcal{N}
\]  

(4.15)  

(4.16)

where the last set of constraints is automatically satisfied by the employed routing algorithm, for \( \forall r \in R(v,s,\sigma) \).

Note that the problem reduces to maximizing a nonlinear objective function constrained with a convex set of linear inequalities. The non-linearity stems from the very complicated dependence of the network loss probabilities \( L_{usr} \), on the offered traffic rates,
4.3 The solution approach

\( \rho_{\text{usr}} \), and the VP links capacities, \( C_l \), for which there is no closed form expression. However, \( W \) and its respective partial derivatives can be evaluated numerically, for fixed values of \( \rho_{\text{usr}} \) and \( C_l \).

There is no single standard solution algorithm which solves this type of problem. Furthermore, since the objective function is neither convex nor concave, the algorithms will only lead to a local optimum. We have chosen an algorithm, which belongs to the family of the sequential approximation algorithms, known as the Frank-Wolfe algorithm (see [HL95] and references therein). The algorithm replaces the nonlinear objective function by a succession of linear approximations. Since the problem is linearly constrained, this substitution allows repeated application of Linear Programming (LP) solution algorithms, accompanied by an one dimensional optimization at each iteration, yielding a sequence of solutions that converges to the optimal solution of the original problem. The algorithm is particularly well suited for linearly constrained problems like the one considered in this subsection.

Let \( g(v,s,r,l) = i \), denote a discrete mapping function that maps the four dimensional space to a one dimensional sequence, and let also \( x = \{x_l\} \) denote a vector variable with arguments defined by

\[
x_{g(v,s,r,l)} = \begin{cases} 
\rho_{\text{usr}} & \text{for } l = 0 \\
C_l & \text{otherwise.}
\end{cases}
\]

The solution algorithm is called the \( \rho C \) Optimizer\(^\text{9} \), and it can then be summarized as follows:

1. Initialize:
   Generate initial solution \( x \)
   \( i \leftarrow 0 \)
   evaluate \( W^{(0)} \)
   not converged \( \leftarrow \) TRUE

2. While (not converged) do

   2.1. Find \( L_{\text{usr}}, \frac{\partial W}{\partial \rho_{\text{usr}}}, \frac{\partial W}{\partial C_l} \) at \( x = x^{(i)} \)

   2.2. Find \( x^{LP} \) (i.e. \( \rho_{\text{usr}}^{LP}, C_l^{LP} \)) by solving:

   Maximize: \( \sum_i \frac{\partial W}{\partial x_i} x_i \)

   Subject to: \( Ax \leq b \)

   2.3. For \( 0 \leq \alpha \leq 1 \), find \( \alpha_{\text{max}} \) that maximizes \( W(x^{(i)} + \alpha(x^{LP} - x^{(i)})) \).

   2.4. \( x^{(i+1)} = x^{(i)} + \alpha_{\text{max}}(x^{LP} - x^{(i)}) \)

\(^9\)The algorithm determines the optimal traffic distribution \( \{\rho_{\text{usr}}^*\} \) and VP capacities \( C^* \), hence its name.
2.5. if \((W(x^{(i+1)}) - W(x^{(i)}))/W(x^{(i)}) \leq \epsilon\) 
not-converged \(\leftarrow\) FALSE

2.6. \(i \leftarrow i + 1\)

end while

3. Return \(\rho_{vss}^{*} \leftarrow x_{g(v,s,s,0)}^{(i)}, \ C_{l}^{*} \leftarrow x_{g(v,s,s,l)}^{(i)}\)

where the inequality \(Ax \leq b\) expresses the constraints (4.5) and (4.15), in a matrix form.

For calculation of the network loss probabilities \(L_{usr}\) in step 2.1, I use the fixed-point approach, which is based on the link independence assumption [Ros95, Kel86]. The equations for the network revenue sensitivities \((\frac{\partial W}{\partial x^{(i)}}, \frac{\partial W}{\partial C_{l}})\) and the implied costs\(^{10}\), are obtained by extension of the approach of Kelly [Kel88, Kel90], in [MMR96]. The extension has also been made independently by Faragó et.al. in [FBA+95]. This is also the most intricate part of the whole algorithm.

Using the results for the partial derivatives of the network revenue, the optimization problem is approximated by a linear programming problem. The LP problem is solved by applying the Simplex method in Step 2.2. Since the original objective function is nonlinear, it may not continue to increase all the way along the line from \(x^{(i)}\) to \(x^{lp}\). Therefore instead of taking \(x^{lp}\) as the new trial solution, step 2.3 tries to find the point that maximizes the original objective function along this line segment, by conducting a one-dimensional search. That point then becomes the new trial solution in step 2.4. The sequence of trial solutions generated by repeated iterations of the algorithm, converges to the local optimum of the original problem. The algorithm stops as soon as the revenues of the successive trial solutions do not change significantly between successive trials.

**Network revenue sensitivity and implied cost analysis**

Since both the network revenue sensitivities and the implied costs are independent of the design variables of the other subnetworks, the analysis in this section focuses on a single VP subnetwork. Therefore I will drop the subscript \(v\) from all relevant variables. The obtained results can be applied to each subnetwork separately. First I describe the reduced load approximation for determination of the network loss probabilities, and then the equations for the network revenue sensitivities and the implied costs. The structure of the subsection is strongly influenced by an excellent work of Mitra et.al. [MMR96, MMR95].

---

\(^{10}\)The implied cost \(c_{k}\) is defined as the expected loss in revenue due to accepting a call of class \(k\) in equilibrium. The formal definitions are given in subsection Network revenue sensitivity and implied cost analysis on page 108. Further details can be found in [Kel86, Kel88, Kel90, Ros95].
4.3 The solution approach

Let $B_{sl}$ denote the blocking probability of service $s$ calls on VP link $l$. Recall from section 4.2.1 that each service $s$ call requires $d_{sl}$ bandwidth units on VP link $l$, where $d_{sl}$ is obtained from the appropriate effective bandwidth approximation. Let $d_l = \{d_{l1}, \ldots, d_{sl}\}$ denote the effective bandwidth vector and $\nu_l = \{\nu_{l1}, \ldots, \nu_{sl}\}$ the reduced load vector, where $\nu_{sl}$ represents the reduced load of service $s$ calls offered to link $l$. Then $B_{sl}$ can be found as

$$B_{sl} = L_s(\nu_l, d_l, C_l),$$

where $C_l$ is the capacity of the VP link $l$, and the function $L_s$ can be calculated exactly by the recursive algorithm of Kaufman and Roberts, for calculation of the blocking probability of a stochastic knapsack [Ros95]. Note that the complexity of calculating $B_{sl}$ by this recursion is $O(C)$ as $C \to \infty$.

Observe that $B_{sl}$ can also be interpreted as the probability that less than $d_{sl}$ bandwidth units are free on link $l$. Suppose that these events are independent from link to link. Then, since the service $s$ calls arrive according to a Poisson process with load $\rho_{sr}$ on each route $r$, the reduced load of service $s$ calls on link $l$ is given by:

$$\nu_{sl} = \sum_{\sigma} \sum_{r \in R(s, \sigma) \mid e \in r} \rho_{sr} \prod_{m \in r - \{l\}} (1 - B_{sm})$$

(4.18)

The last two expressions, (4.17) and (4.18), constitute the reduced load approximation, also known as the Erlang fixed-point approximation. The approximation procedure can be summarized as follows:

$$B_{sl} = \phi_{sl}(\nu_l),$$

$$\nu = \Psi(B),$$

(4.19)

where $\nu = \{\nu_{sl}\}_{s,l}$ and $B = \{B_{sl}\}_{s,l}$, $S$ denotes the subset of services supported by subnetwork $v$, and $L$ denotes the subset of VP links belonging to subnetwork $v$. The function $\phi_{sl}$ is obtained from (4.17) by fixing the $d_l$ and $C_l$, which is why (4.19) are also called the fixed-point equations, and $\Psi$ is defined by (4.18).

The link blocking probabilities $B_{sl}$, are typically calculated by employing the method of successive substitutions to (4.19). Having done that, and taking into account that blocking is independent from link to link, the loss probability of service $s$ on route $r$, $L_{sr}$, can now be approximated with

$$L_{sr} \approx 1 - \prod_{l \in r} (1 - B_{sl}).$$

(4.20)

The network revenue sensitivities are closely related to the concept of implied costs. Each call that is admitted to the network will generate expected revenue proportional to its holding time. But the call will also cause a loss in the future expected revenue due to the additional blocking that its presence causes. The expected loss in revenue $c_{sl}$, due to the

---

More efficient algorithms for calculation of the blocking probability of the stochastic knapsack are also available, see for instance [AvdV96] and the references therein.
removal of \(d_{ul}\) bandwidth units from the link \(l\), is called the implied cost of a service \(s\) call on a link \(l\). The equations for both the network revenue sensitivities and the implied costs, are obtained by extending the approach of Kelly [Kel88, Kel90] to the multirate case. The proofs of the propositions and further derivation details, can be found in [MMR96]. Here I present only the final results.

**Proposition 4.1:** The sensitivity of the network revenue \(W\), with respect to the offered load \(\rho_{sr}\), when the revenue is calculated from the fixed-point equation set, is given by:

\[
\frac{\partial W}{\partial \rho_{sr}} = (1 - L_{sr}) \left( e_{sr} - \sum_{l \in r} c_{sl} \right). \tag{4.21}
\]

For the expression of the implied cost, we also need the following notation. Let \(\nu_{sl,r}\) denote the reduced load of service \(s\) on route \(r\), which is offered to link \(l\), i.e.,

\[
\nu_{sl,r} = \rho_{sr} \prod_{m \in r - \{l\}} (1 - B_{sm}). \tag{4.22}
\]

Note that from (4.18), it follows that the reduced load of service \(s\) offered to link \(l\) is:

\[
\nu_{sl} = \sum_{r} \sum_{r \in R(s, \sigma)} \nu_{sl,r}. \tag{4.23}
\]

Then the implied costs are defined with the following set of \(SL\) linear equations:

\[
c_{ul} = \frac{1}{(1 - B_{ul})} \sum_{\sigma,s} \sum_{r \in R(s, \sigma)} \frac{\partial B_{sl}}{\partial \nu_{ul}} \nu_{sl,r} \left( e_{sr} - \sum_{k \in r - \{l\}} c_{sk} \right), \tag{4.24}
\]

where the expression for \(\frac{\partial B_{sl}}{\partial \nu_{ul}}\) is found by the following proposition [MMR96].

**Proposition 4.2:**

\[
\frac{\partial B_{sl}}{\partial \nu_{ul}} = (1 - B_{ul})[L_s(\nu_l, d_l, C_l - d_u) - L_s(\nu_l, d_l, C_l)]. \tag{4.25}
\]

We also need the network revenue sensitivity with respect to the link capacity. In [MMR96] it was also shown that the network revenue sensitivity with respect to the link capacity, can be easily found from the implied costs by:

\[
c_{ul} = d_{ul} \frac{\partial W}{\partial C_l}, \tag{4.26}
\]

To solve the equations for the implied costs, one needs the solution of the fixed-point solution (4.19), in order to determine the values of \(\{B_{ul}\}, \{\partial B_{sl}/\partial \nu_{ul}\}\) and \(\{\nu_{sl,r}\}\). Note that the complexity of solving (4.24) is \(O(S^3L^3)\). Since the number of supported services is expected to be large, and taking into account the high modularity of the network links which impacts the computational complexity of (4.17), it is clear that simplifications are necessary in order to achieve reasonable computation times.
4.3 The solution approach

In order to reduce the computational complexity I apply the uniform asymptotic approximation (UAA), developed by Mitra and Morrison [MM94], for the analysis of a single link. The approximation is included in various stages of the analysis, yielding numerical procedures with complexity which is independent of $C$ and $S$. Further details about the applicability and the accuracy of the UAA approach, can be found in [MM94, MMR95, MMR96]. Here I present only the results that are necessary for the operation of the solution algorithm.

The asymptotic regime for the UAA is large $C$ and large offered traffic, i.e. $C \gg 1$ and $\nu_{sl} = O(C)$, where $C$ is an integer. It is further assumed that the $d_{sl}$s are positive integers, not large relative to $C$, with the greatest common divisor $1^{12}$. Let $F_l(z)$ and $V_l(z)$ denote two functions defined as

$$F_l(z) = \sum_{s=1}^{S} \nu_{sl}(z^{d_{sl}} - 1) - C \log z$$

$$V_l(z) = \sum_{s=1}^{S} d_{sl}^{2} \nu_{sl} z^{d_{sl}}$$

(4.27)  

(4.28)

Let $z_l^*$ denote the unique positive solution of $F_l'(z) = 0$, where the prime denotes the first derivative of $F_l(z)$. Then, let

$$b_{sl} = \frac{\left[ \frac{1-\left(z_l^*ight)^{d_{sl}}}{(1-z_l^*)} \right]}{d_{sl}} z_l^* \neq 1$$

$$b_{sl} = 1$$

(4.29)

$$B_l = \frac{e^{F_l(z_l^*)}}{G_l \sqrt{2\pi V_l(z_l^*)}}$$

(4.30)

where

$$G_l = \left\{ \begin{array}{ll} \frac{1}{2} \text{Erfc}[\text{sgn}(1 - z_l^*) \sqrt{-F_l(z_l^*)}] + \frac{e^{F_l(z_l^*)}}{2\pi} \left[ \frac{1}{V_l(z_l^*)(1-z_l^*)} - \frac{\text{sgn}(1-z_l^*)}{\sqrt{-2F_l(z_l^*)}} \right] & z_l^* \neq 1 \\ \frac{1}{2} \left[ 1 + \frac{1}{\sqrt{2\pi V_l(z_l^*)}} \left[ 1 + \frac{1}{3V_l(z_l^*)} \sum_{s=1}^{S} d_{sl}^{2} \nu_{sl} \right] \right] & z_l^* = 1 \end{array} \right.$$

The complementary error function and the signum function, are defined in the standard way:

$$\text{Erfc}(y) = \frac{2}{\sqrt{\pi}} \int_{y}^{\infty} e^{-x^2} dx$$

$$\text{sgn}(x) = \begin{cases} \frac{x}{|x|} & x \neq 0 \\ 1 & x = 0 \end{cases}$$

(4.31)  

(4.32)

Then the blocking of service $s$ calls on link $l$ can be approximated by [MMR96, MMR95]:

$$B_{sl} \approx b_{sl} B_l$$

(4.33)

$^{12}$The condition is easily satisfied by normalizing the $d_{sl}$s. If some of the $d_{sl}$s are zero, then a slightly adapted procedure applies (for details see [MMR95, MMR96]).
The approximation holds irrespective of whether the link is in the overloaded \((0 < z_i^* < 1)\), critical \((z_i^* \approx 1)\) or underloaded \((z_i^* > 1)\) regime. Note that the numerical complexity of calculating \(B_{si}\) is \(O(1)\), which is a great improvement with respect to (4.17). Furthermore, since the successive substitutions method which is used for solving the fixed-point equations (4.19) requires many evaluations of \(B_{si}\), by substituting equation (4.17) with equation (4.33), one can achieve substantial reduction of the numerical burden.

UAA analysis can also be used to further reduce the complexity of calculating the implied costs, \(c_{si}\). Namely, in [MMR95, MMR96] it has been shown that

\[
L_s(\nu_t, d_t, C_t - d_t) - L_d(\nu_t, d_t, C_t) = \frac{1}{(1 - B_{ul})} \frac{\partial B_{si}}{\partial \nu_t} \approx a_{ul} B_{si} \tag{4.34}
\]

where

\[
a_{ul} = \frac{(z_i^*)^{d_{ul}}}{(1 - B_{ul})} - 1. \tag{4.35}
\]

By substituting expression (4.34) in (4.24), one obtains

\[
c_{ul} = a_{ul} \sum_{\sigma, \tau} \sum_{r \in R(s, \sigma), l \in r} \nu_{srl} B_{si} \left( e_{sr} - \sum_{k \in r \setminus \{l\}} c_{sk} \right). \tag{4.36}
\]

Now define \(\xi_l\) as

\[
\xi_l = \sum_{s, \sigma, \tau} \sum_{r \in R(s, \sigma), l \in r} \nu_{srl} B_{si} \left( e_{sr} - \sum_{k \in r \setminus \{l\}} c_{sk} \right). \tag{4.37}
\]

Note that for the feasible values of \(t (1 \leq t \leq S)\) and \(l (1 \leq l \leq L)\), the implied costs become:

\[
c_{ul} = a_{ul} \xi_l. \tag{4.38}
\]

Now by substituting (4.38) into (4.37), one obtains

\[
\xi_l = \sum_{s, \sigma} \sum_{r \in R(s, \sigma), l \in r} \nu_{srl} B_{si} \left( e_{sr} - \sum_{k \in r \setminus \{l\}} a_{sk} \xi_k \right). \tag{4.39}
\]

Observe that (4.39) is a complete set of equations in \(\{\xi_l\}\). Remarkably (4.39) is a system of only \(L\) equations. Once it is solved, the implied costs can be easily obtained from (4.38). Thus the numerical complexity of obtaining the implied costs has been reduced to \(O(L^3)\). This is a direct consequence of the incorporation of the UAA approach in the network analysis.
Typically not all VP links in the subnetwork will be large and thus subject to UAA analysis. Therefore the pC Optimizer algorithm employs a hybrid system of equations, that combine both the asymptotic and non asymptotic regimes. The equations are easily obtained from the previous expressions.

Let $\mathcal{L}_A$ denote a subset of links in the subnetwork, which are amenable to UAA analysis, and let $\mathcal{L}_E$ denote the subset of subnetwork links, which are not amenable to UAA analysis. For the calculation of the link blocking probabilities $B_{sl}$, one obviously uses the appropriate method, for $l \in \mathcal{L}_A$ equation (4.33), and for $l \in \mathcal{L}_E$ equation (4.17). The successive approximation method applies in a straightforward manner. For calculation of the implied costs, the equations must be slightly adapted. From (4.37) and (4.38), for $l \in \mathcal{L}_A$, one obtains:

$$\xi_l = \sum_{s, \sigma} \sum_{r \in \mathcal{R}(s, \sigma)} \nu_{sl;r} B_{sl} \left[ e_{sr} - \sum_{k \in \mathcal{L}_E \cap (r-\{l\})} c_{sk} - \sum_{k \in \mathcal{L}_A \cap (r-\{l\})} a_{sk} \xi_k \right]$$

(4.40)

and $c_{dl} = a_{dl} \xi_l$. For $l \in \mathcal{L}_E$, from (4.24) and (4.25), one obtains

$$c_{dl} = \sum_{s, \sigma} \sum_{r \in \mathcal{R}(s, \sigma)} [L_s(\nu_l, d_l, C_l - d_l) - L_s(\nu_l, d_l, C_l)] \nu_{sl;r}$$

$$\times \left[ e_{sr} - \sum_{k \in \mathcal{L}_E \cap (r-\{l\})} c_{sk} - \sum_{k \in \mathcal{L}_A \cap (r-\{l\})} a_{sk} \xi_k \right].$$

(4.41)

Note that (4.40) and (4.41) form a complete system of equations in $\xi_l$ ($l \in \mathcal{L}_A$) and $c_{sl}$ ($l \in \mathcal{L}_E; s \in S_v$).

### 4.4 BANDIT: a flexible ATM network design tool

In order to investigate the quality of the design framework described in the previous sections, we\textsuperscript{13} developed a software tool that implements the design framework, called BANDIT (flexible ATM Network Design Tool). In addition, the tool should also show the feasibility of the framework and the employed algorithms. The choice of a software tool for the analysis is dictated by the problem complexity and inter-layer dependencies, which makes it very unsuitable for any type of analytical analysis approach.

There are two essential requirements that the tool must satisfy, 
\textit{user-friendliness} and \textit{flexibility}. In other words, BANDIT must be easy to use and easy to change. User-friendliness is a natural requirement for any software product. In our context ease of use primarily means to be able to input/view data graphically and to control the whole design process through a Graphical User Interface (GUI). The flexibility requirement is driven by the nature of the tool’s usability. Namely, BANDIT is primarily used in a research environment as a testing tool. Thus the ability to easily incorporate various subproblem solution algorithms within the design framework, is a valuable feature. By extension, the ability to easily add additional functionality is very important for further research activities.

\textsuperscript{13}BANDIT has been developed in close collaboration with G. Klop.
Based on the design requirements and the time frame available for the realisation of
the project, BANDIT was developed using an Object Oriented Programming (OOP) ap-
proach, and written partly in C++ and partly in Tcl [Ous94, Wel95]. The OOP allows for
development of flexible and easily extendible applications, whereas Tcl allows for fast de-
velopment of the graphical elements of the application. Since Tcl is a scripting language,
and therefore interpreted at run-time, it has a negative impact on the tool's performance.
This however does not represent a problem, since virtually all computationally expen-
sive design procedures are written in C++, which boosts the performance significantly,
whereas the Tcl scripts basically control the appearance of the tool.

In general, BANDIT consists of two basic components, the GUI and the design engine.
The GUI allows for graphical interaction between the user and the application. The de-
sign engine on the other hand implements the core functionality of the application,
based on the design framework discussed in subsection 4.3.1. Since the design framework
was discussed in great detail in the previous sections, the primary focus of this section is
the GUI of BANDIT.

4.4.1 The Graphical User Interface

The primary task of the tool's GUI is to provide graphical means to control the whole
network design process. This includes input of static design data, visual presentation of
the design process and the obtained results, generation of reports and other necessary
housekeeping activities. This is achieved through a number of functional windows, like
the desktop, various editor and manager windows, each performing a specific task. Hence
the rest of the subsection describes the various functional windows that constitute the
GUI of BANDIT.

The Desktop window of BANDIT (see figure 4.3), is the main control window of the
application. It provides visual feedback before, during and after the design cycle, as well
as full user control over the design process itself. For instance, from the desktop one can
import a sample network configuration, specify the design parameters, start and stop the
design process, publish the obtained results or simply consult various help topics. The
desktop is also the parent window of all other functional windows.

The appearance of the desktop window is determined by the amount of static data
present in the data structures\textsuperscript{14}. The behavior is determined by a Finite State Machine
(FSM), see figure 4.4. The FSM consists of the following six states:

Def.Net – This is the default state when one begins a new design process. There is no
static data present in the data structures, hence the desktop is empty. The first
step is to define the PHY layer of the ATM network, i.e. the network nodes and the
physical topology. This is done by invoking the Network Editor.

\textsuperscript{14}Because BANDIT is written using an OOP approach the data is contained in various objects, which
are created and destroyed in the course of the design process. The classical meaning of the term data
structure is therefore not completely correct. Nevertheless it is used in the text for simplicity reasons
and it refers to the loose set of objects that carry the data.
4.4 BANDIT: a flexible ATM network design tool

Figure 4.3: A sample desktop window of BANDIT.

Figure 4.4: The Finite State Machine diagram of BANDIT.

**Def.Traffic** – This state is entered when the only static data present in the data structures is the specification of the physical ATM network, which is also shown in the desktop window. The next step is to define the reference traffic parameters, like traffic intensity, QoS requirements etc. This is done in the Traffic Editor.

**Def.Design** – This state is entered when both the physical network and the reference
traffic parameters have been defined. Here the user must specify the parameters of
the design procedure, like the subproblem solution algorithms and their parameters
etc. This is done in the Design Editor.

Ready_To_Optimize – This state is entered when all static data is already present in
the data structures. This is also the starting and the ending state of the optimization
process. The desktop shows the specified (sub)network topology and all relevant
design measures, like call blocking, network costs etc.

Running – In this state BANDIT is searching for the optimal VP network configuration.
The desktop shows the evolution of the design process. The user can follow the
changes in the topology of various VP subnetworks, the link capacities, etc. If
stopped the application returns to the Ready_To_Optimize state.

Paused – This state is entered only when the design process is interrupted, in order
to investigate the intermediate results. No changes to the data are allowed in
this state. The application can either return to the Running or jump to the
Ready_To_Optimize state depending on the user’s actions.

All permitted state transitions are depicted in figure 4.4. When importing a network
configuration from a file, the application enters one of the first four states, according
to the amount of information available in the input file. The interstate transitions are
achieved by employing the respective editor. BANDIT comprises several editors, the
Network Editor, the Traffic Editor, the Design Editor and the Initial nets Editor, each
one performing a respective function. All editors are invoked through the desktop window.

The Network Editor is depicted in figure 4.5. Its main function is to define the physical
layer topology of the ATM network. For that purpose it contains functional buttons for
creating ATM nodes and SONET/SDH links between them. The user can also specify the
node and link attributes, like node names, link capacities etc. Before exiting the Network
Editor checks the consistency of the input data, like network connectivity, parameter
ranges and similar tests.

The Traffic Editor is depicted in figure 4.6. As its name suggests, the Traffic Editor
is concerned with the description of the reference traffic. It consists of four tabs (see
figure 4.6), Services, Traffic Integration, Traffic intensity and Avg Number of Calls tab,
describing CELL level traffic service parameters, traffic integration policy, CONNECTION
level traffic intensity and average number of call arrivals per service, respectively.

The Services tab is used to define all service classes that should be supported by
the network15. A service class is modeled as a multi-state continuous Markov chain,
characterized by a rate in each state and a transition probability matrix. By setting the
number of states to two, the model becomes the familiar ON-OFF model described in
chapter 2. Figure 4.6 gives an example of the Services tab, with all necessary parameters.
Note that the Services tab specifies the CELL level traffic modeling, which is necessary for
the calculation of the effective bandwidth.

15Services are distinguished according to their QoS requirements (both loss and delay requirements), i.e.
a service class in BANDIT corresponds to a traffic type in the network model defined in subsection 4.2.1.
4.4 BANDIT: a flexible ATM network design tool

Figure 4.5: A sample screen of the Network Editor. The editor shows the topology of the experimental physical network.

Figure 4.6: Two sample tab screens of the Traffic Editor.

The Traffic Integration tab specifies how the network resources are shared between the various services. As already discussed in section 4.1, there are three different approaches for resource sharing (or statistical multiplexing) in ATM networks. Namely Complete Sharing (CS), Partial Sharing (PS) and Complete Partitioning (CP) (see [RMV96]). In the Traffic Integration tab the user can specify which approach should be used during the optimization part. In addition, in the case of PS, the user must also specify the distribution of services between the various VP subnetworks, i.e. which services are supported by
each subnetwork.

The last two tabs specify \texttt{CONNECTION} level traffic parameters. Namely the traffic intensity and the average number of calls, per source-destination pair $\sigma$ and per service $s$.

The \textit{Design Editor} is depicted in figure 4.7. It determines the operation mode of the design engine as well as the parameters of the employed design algorithms. We distinguish four different operation modes: long-term design, medium-term design, short-term design and testing mode (indicated as \textit{Performance} in figure 4.7) [VHW96].

Figure 4.7: A sample screen of the Design Editor, when the long-term design mode is specified.

The \textit{long-term} design mode assumes no existing network configuration, i.e. it starts the design process from scratch. Both the \textit{medium-term} and \textit{short-term} design modes, assume an already existing VP subnetwork configuration. In \textit{short-term} design mode only the VP link capacities are changed, the topologies remain fixed, whereas in \textit{medium-term} design mode changes to both the topologies and the VP capacities occur locally, as a result of major traffic distribution changes or link failures.

The testing mode is used to investigate the performance of new algorithms for various subproblems. In other words, in testing mode the application solves only a single subproblem, specified by the user. This significantly reduces the development cycle of a new algorithm since the user needs to focus only on solving the specific subproblem, all the supporting structure is already available.
4.5 Preliminary experimental results

Currently only two modes of operation are implemented in BANDIT, the long-term design mode and the testing mode. The development and the implementation of the other two modes is left for future study.

After specifying the operation mode, the Design Editor shows the appropriate design framework. Figure 4.7 depicts the case of the long-term design mode. As expected the framework depicted in the figure, closely follows the design framework described in subsection 4.3.1. Consequently, the user can choose different algorithms for each block, i.e. for each subproblem, and then specify its initial and control parameters in the lower-left subwindow. Thus BANDIT inherits the flexibility built into the design framework, allowing the user to choose among a number of solutions and compare their relative performance. By extension, BANDIT can be easily extended to incorporate additional technological developments, like priority queuing mechanisms, as algorithms that can accommodate them become available.

The *Initial nets Editor* is invoked before a new design loop is started. As the name suggests, its only task is to define the initial VP subnetwork topologies. The initial topologies can be defined by the user himself or the application can generate random topologies in line with certain user specified parameters.

Beside the editors, another category of functional windows which are also available in BANDIT are the so-called managers. Four managers are currently implemented: the options manager, the report manager, the print manager and the help manager.

The options manager controls several aspects of the way how the application appears and functions. Examples are specifying default directories, colors, fonts, design process options etc. The report manager presents the results of the design process, organized by topics of interest, specified by the user. The user can also define additional topics of interest, by manually editing the design files, prior to the design process. The print and the help managers are performing the typical tasks of printing various windows and providing guiding instructions to the new user.

Both the tool and the experimental results discussed in the following section, were presented at the MASCOTS'98 conference. For more detailed information about BANDIT, including the implementation details, the reader is referred to the conference proceedings [KS98] (see also [Klo98]).

4.5 Preliminary experimental results

This section discusses the results of the preliminary experiments obtained by employing the design framework and the software tool. Due to time limitations, the section does not include an in-depth sensitivity analysis of the obtained results nor a detailed performance testing of the tool. Instead the main aim of the section is to demonstrate the applicability of the design framework and to provide some indication of the quality of the obtained results.
Chapter 4. ATM network design

The experiments involve a single physical ATM network layer, with predefined traffic demands and QoS requirements, and a VP subnetwork configuration which is subsequently optimized according to various resource sharing strategies. The next subsection describes the physical network structure and the input traffic parameters. The last subsection discusses the results of the design process.

4.5.1 Network definition

Since the network used in the experiments is a fictitious regional network in the Netherlands, all the parameters are imaginary. In order to simplify the presentation, the parameters are divided into physical network parameters, traffic parameters and design parameters, in a similar fashion as they are inserted into BANDIT, through the respective editor.

The physical network topology together with the related parameters, is depicted in figure 4.5. The network consists of 6 nodes connected by 16 unidirectional links. All links have a capacity of 45 Mbit/s, with exception of the links between nodes 1–3 (Delft–Rotterdam) and 2–3 (Gouda–Rotterdam) which have a capacity of 90 Mbit/s. Since the bandwidth granularity of BANDIT was set to 16 Kbit/s, the link capacities of 45 and 90 Mbit/s become 2812 and 5625 capacity units, respectively.

Following the approach of Chapter 2, the network traffic is classified into LX, LD, LL and HX traffic class, according to its loss and delay requirements. The relevant traffic parameters include BURST level, CONNECTION level and design parameters.

<table>
<thead>
<tr>
<th>Service</th>
<th>Class</th>
<th>$R_{peak}$ [16 kbit/s]</th>
<th>$1/\theta$ [sec]</th>
<th>$\gamma_{\theta + \eta}$</th>
<th>$\epsilon_s$ – loss probability</th>
<th>cell delay [VP hops]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video</td>
<td>LX</td>
<td>40</td>
<td>10</td>
<td>0.889</td>
<td>$10^{-9}$</td>
<td>2</td>
</tr>
<tr>
<td>Voice</td>
<td>LD</td>
<td>4</td>
<td>1.67</td>
<td>0.333</td>
<td>$10^{-3}$</td>
<td>2</td>
</tr>
<tr>
<td>Data1</td>
<td>LL</td>
<td>24</td>
<td>1.25</td>
<td>0.238</td>
<td>$10^{-6}$</td>
<td>4</td>
</tr>
<tr>
<td>Data2</td>
<td>HX</td>
<td>6</td>
<td>2.17</td>
<td>0.578</td>
<td>$10^{-3}$</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 4.1: ON-OFF traffic model parameters and respective QoS requirements

The burst level parameters are given in table 4.1. They correspond to the parameters of the ON-OFF traffic model defined in section 2.3.2, and the QoS requirements of the traffic source. The delay requirement determines the size of the routing set $R(s, \sigma)$ in the Routing and Dimensioning step, whereas all other parameters are used for the computation of the effective bandwidth. Note that the choice of the effective bandwidth approximation is not essential to the operation of the design procedure. The approximation used in the experiments is given with the following expression [GAN91, RMV96]:

$$d_{st} = \frac{\eta_s + \theta_s + R_{peak}^{(s)} \cdot z_0 - \sqrt{(\eta_s + \theta_s + R_{peak}^{(s)} \cdot z_0)^2 - 4 \eta_s R_{peak}^{(s)} \cdot z_0}}{2z_0}$$  (4.42)
where \( z_0 = \ln \epsilon / x_t \), \( x_l \) is the size of the output buffer, whereas \( \epsilon = \min \{ \epsilon_s \} \) for all services clustered in the same VP subnetwork.

The \textit{connection} level parameters are depicted in tables 4.2 through 4.6. Tables 4.2 through 4.5 represent the traffic intensity matrices \( \{ p_{sr} \} \), for the LX, LD, LL and HX traffic class, respectively. Table 4.6 on the other hand, specifies the average call duration, \( 1/\mu \), per class. Note that based on the information in the tables one can easily determine the average number of calls per second, \( \lambda \).

<table>
<thead>
<tr>
<th>Node</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>27</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>31</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>10</td>
<td>0</td>
<td>23</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.2: LX traffic intensity matrix.

<table>
<thead>
<tr>
<th>Node</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>11</td>
<td>6</td>
<td>15</td>
<td>3</td>
<td>34</td>
</tr>
<tr>
<td>2</td>
<td>18</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>13</td>
<td>0</td>
<td>23</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>28</td>
<td>15</td>
<td>5</td>
<td>0</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>6</td>
<td>4</td>
<td>17</td>
<td>0</td>
<td>11</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>17</td>
<td>8</td>
<td>19</td>
<td>9</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.3: LD traffic intensity matrix.

<table>
<thead>
<tr>
<th>Node</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>18</td>
<td>15</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>0</td>
<td>2</td>
<td>10</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>11</td>
<td>0</td>
<td>21</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>25</td>
<td>9</td>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.4: LL traffic intensity matrix.

<table>
<thead>
<tr>
<th>Node</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>11</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>0</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>18</td>
<td>24</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.5: HX traffic intensity matrix.

<table>
<thead>
<tr>
<th>Service</th>
<th>Video</th>
<th>Voice</th>
<th>Data1</th>
<th>Data2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call duration [sec]</td>
<td>1800</td>
<td>180</td>
<td>10</td>
<td>50</td>
</tr>
</tbody>
</table>

Table 4.6: The average call duration \( (1/\mu) \) per service class.

The design parameters are grouped according to the subnetwork solution algorithm they belong to. The parameters of the \textit{Simulated Annealing} algorithm (Topology Optimization step) are as follows: \textit{INITTEMP} = 5, \textit{SIZEFACTOR} = 8, \textit{TDelay} = 0.95, \textit{CUTOFF} = 0.4, \textit{MINPERCENT} = 2\%, \textit{FROZEN} = 3 and \textit{CHANGE} = 0.7. The last parameter is particular to this implementation and it specifies the probability that when generating a new configuration, besides exchanging the links of the two randomly chosen nodes, the algorithm also changes the connection state between the two nodes. If they are connected, the link is removed and vice versa.

The parameters of the \textit{pC Optimizer} algorithm (Routing and Dimensioning step) are given in table 4.7. The \textit{NrHops} and the \textit{Plus} parameters determine the size of the routing
set $\mathcal{R}(v, s, \sigma)$. Since the network model employs static routing, the traffic stream $(v, s, \sigma)$ is uniformly spread over all routes that belong to $\mathcal{R}(v, s, \sigma)$, where $\mathcal{R}(v, s, \sigma)$ is defined as:

$$\mathcal{R}(v, s, \sigma) = \{ r | \text{hops}(r) \leq \min(\text{min-hop}(\sigma) + \text{Plus}, \text{NrHops}) \} \quad (4.43)$$

where $\text{hops}(r)$ function returns the number of VP hops on route $r$ and the $\text{min-hop}(\sigma)$ function returns the number of hops on the shortest path route for the $\sigma$ source destination pair. Note that the $\text{NrHops}$ basically corresponds to the delay requirement of the respective traffic class as defined in table 4.1. If the VP subnetwork topology is such that certain routing sets are empty, then that particular traffic is routed through the shortest path route and a large penalty is added to the overall network cost. Such an approach improves the performance of the Simulated Annealing algorithm.

### 4.5.2 Experimental results

This subsection discusses the results of the three experiments performed with the sample network described above. The objective of the first experiment is to provide some indication of the quality of the design framework. The discussion also suggests possible techniques for obtaining bounds on the optimal solution. The other two experiments illustrate the applicability of the design framework and the impact of the ATM multilayer structure on the network efficiency, by considering two different resource sharing strategies. The emphasis is on performing qualitative comparison between the different resource sharing options. The quantitative analysis is left for further study.

The first experiment employs the Complete Partitioning (CP) sharing strategy, i.e. each traffic class is assigned a separate VP subnetwork. The corresponding traffic sharing matrix is given in table 4.8. The optimal VP subnetwork topologies are depicted in figure 4.8, and the overall network cost is given in table 4.11.

<table>
<thead>
<tr>
<th>Service</th>
<th>Video</th>
<th>Voice</th>
<th>Data1</th>
<th>Data2</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPnet1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>VPnet2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>VPnet3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>VPnet4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4.8: Traffic sharing matrix for the CP case.
4.5 Preliminary experimental results

Figure 4.8: The topologies of the VP subnetworks of the optimal network configuration in the CP case.

Because of the problem complexity, it is very difficult to analyze the optimality of the obtained solutions. However since the design framework is heuristic in nature, a good indication of the quality of the solutions, can be obtained by observing the course of the overall network cost during the optimization process. Because the convergence of the design procedure is controlled by the Simulated Annealing algorithm, the network cost curve should experience two characteristics. One, it should be decreasing, but it should also contain random increasing jumps. This is a consequence of the nature of the annealing algorithm that occasionally also accepts more expensive solutions. Second, the jumps should become smaller in size as the temperature decreases, until finally the new solutions stop bringing any significant cost improvements. This is because the probability of accepting a more expensive solution is proportional to the temperature parameter.

The network cost curve is depicted in figure 4.9. Its shape is in very good agreement with our expectations, which indicates that the design framework is working properly. One might argue that this rather intuitive quality measure confirms only the proper operation of the Simulated Annealing algorithm. Note however that if any of the algorithms does not perform properly the network cost curve would be a much more volatile, and more importantly on average it would remain more or less on the same level. In other words, only the size of jumps would decrease with the temperature, not the average network cost. None of the performed experiments produced a network cost curve that exhibits such an
erratic behavior. Not observing this is of course not a full proof, but a good indication that all subnetwork algorithms, as well as the design framework as a whole, are working properly.

Figure 4.9: The overall network cost development in the course of the Simulated Annealing, for the CP strategy.

Of course working properly does not necessarily mean working optimally. In order to investigate the optimality of the design framework and the underlying algorithms more closely, it is highly desirable to have some quantitative figures for comparison, e.g. bounds. Mainly due to time limitations, such results are not included in this thesis. Nevertheless this subsection describes one possible approach for obtaining bounds on the solution of the $\rho C$ Optimizer algorithm (Routing and Dimensioning step).

A bound on the solution produced by $\rho C$ Optimizer algorithm, could be obtained by approximating the traffic processes with deterministic flow processes, with uniform rates over time. In this case the problem becomes a general linear programming problem\textsuperscript{16}. It belongs to the class of multi-commodity maximization problems, for which of the shelf solution algorithms are readily available. Intuitively, the optimal solution of the linear problem must always be smaller or equal to the optimal solution of the original problem. It should also be possible to formally proof the last statement.

The remaining two experiments, employ the partial sharing (PS) strategy. Experiment 2 combines the LD and the LL classes into one subnetwork and has a separate subnetwork for both LX and HX classes. Experiment 3 on the other hand, combines the LD and the HX classes and treats the LX and the LL class separately. The resource sharing matrices are given in tables 4.9 and 4.10. The QoS requirements of the cluster subnetwork correspond

\textsuperscript{16}Recall that the original problem had a non-linear objective function, constrained with a convex set of linear inequalities.
### 4.5 Preliminary experimental results

<table>
<thead>
<tr>
<th>Service</th>
<th>Video</th>
<th>Voice</th>
<th>Data1</th>
<th>Data2</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPnet1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>VPnet2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>VPnet3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4.9: Traffic sharing matrix for the PS1 case.

<table>
<thead>
<tr>
<th>Service</th>
<th>Video</th>
<th>Voice</th>
<th>Data1</th>
<th>Data2</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPnet1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>VPnet2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>VPnet3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.10: Traffic sharing matrix for the PS2 case.

to the most stringent requirements of the combined classes. For instance in experiment 2, the maximal allowed loss probability and delay in the VPnet2, are \(10^{-7}\) and 2 hops, respectively.

The objective of the experiments is to demonstrate the versatility of the design framework in employing various resource sharing strategies in the design process, and to illustrate the impact of the VP subnetwork concept on the efficiency of the ATM network.

The optimal topologies of the VP subnetworks, obtained from experiments 2 and 3, are given in figures 4.10 and 4.11, respectively. Note that the topologies of the corresponding subnetworks are different in all three experiments. This might seem strange at first, especially for the VPnet1 subnetwork carrying only LX traffic. However, recall that in the experiments, all VP subnetworks are mapped onto the same physical network, and therefore are sharing the same physical capacity. Hence changes in one of the subnetworks, will ultimately affect all subnetworks.

Table 4.11 depicts the overall network costs of the results of the three experiments. The table indicates that the best resource sharing strategy in our case, is the CP strategy because it incurs the minimal costs for the same level of performance. This is of course not surprising since the combined classes have higher effective bandwidth in the PS case than in CP case, because of the provision of the most stringent QoS requirements. Consequently both the control and the blocking costs are higher. This illustrates the impact of the VP subnetwork concept.

<table>
<thead>
<tr>
<th>Resource sharing strategy</th>
<th>CP</th>
<th>PS1</th>
<th>PS2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall network cost</td>
<td>15424.38</td>
<td>23621.68</td>
<td>19095.58</td>
</tr>
</tbody>
</table>

Table 4.11: Optimal overall network costs of the three experiments.

Although this line of thinking suggests that CP strategy would be always the best resource sharing strategy, this is not necessarily the case. Recall that the analysis in Chapter 3, has shown that the performance of both classes is improved when the LDOLL queue is used. Hence by employing the LDOLL queuing in the network nodes, the PS strategy might achieve better results than the CP strategy. Furthermore note that the classes discussed here are distinguished according to their QoS requirements. One can
easily define distinct traffic classes, that have the same QoS requirements, but other distinguishing traffic parameters. From that perspective the CP strategy considered here actually corresponds to a PS strategy.

In general all experiments indicate that the design framework is working properly. The optimality of the obtained solutions needs further investigation. The framework can also be easily applied to different resource sharing strategies. The analysis has shown that in the current settings, the best operating strategy is the CP strategy. This might change if the LDOLL queuing mechanism is employed at the cell level. Before such a comparison could take place, one needs to develop an effective bandwidth approximation for the LDOLL queue.

### 4.6 Conclusion

This chapter is concerned with the design of ATM networks. Several new aspects have been recognized that significantly increase the complexity of the design process. The chapter focuses on the impact of the ATM multilayer structure on the efficiency of the network. In particular, it considers configuring an optimal set of VP subnetworks on top of a physical network infrastructure that minimizes the network cost with respect to specific traffic demands, performance requirements, and other constraints. The objective is to develop a solution procedure that captures all important design aspects and tradeoffs, and it can be easily adapted to include additional functionality in the future (e.g. priority queuing mechanisms).
4.6 Conclusion

The design problem is formulated as nonlinear, mixed integer program with linear constraints. Because of its complexity, the solution procedure uses a heuristic solution approach, based on a design framework that iteratively solves a sequence of relevant subproblems. The design framework is a new tool that can employ different set of algorithms for each of the subproblems [MMR96, FBA+95], and it can analyze a wide range of objective functions. Its flexibility follows from the modularity of the framework, i.e. changes in one subproblem of the framework have minor effect on the other subproblems.

In order to show its feasibility, the proposed framework is implemented into a software tool called BANDIT. Section 4.4 describes the various capabilities and features of BANDIT. Finally section 4.5 discusses several design experiments that illustrate the quality and the flexibility of the design framework.
Chapter 5

Connectionless enterprise ATM overlay network design

All previous chapters have considered general ATM performance and design issues. This chapter however, is concerned with a particular issue of designing a connectionless enterprise network on top of an ATM network. The need for enterprise or private networking stems primarily from the demand for interconnecting the geographically distant local and metropolitan area networks (LANs and MANs, respectively), into a large virtual LAN, allowing a global company to operate more effectively. From a transport point of view, the enterprise network may consist simply of a set of links between a collection of user nodes, or alternatively the links may terminate into public nodes allowing for more efficient sharing of resources. The later approach was taken in the previous chapter, here I discuss the first alternative.

The main challenge stems from the fact that ATM is a connection-oriented, whereas virtually all of the current LANs and MANs are inherently connectionless. An ATM session requires a connection setup phase, prior to the transmission of any ATM cells. On the other hand, in the existing local area networks, like Ethernet, Token Ring, FDDI etc., there is no connection set-up phase and a new session simply starts forwarding packets using common MAC protocols.

According to the ITU-T, there are basically two methods for provision of connectionless service in an ATM network: an indirect and a direct method (see [Tel93c, VS94]). In the indirect method, the connectionless service is realized through the use of switched or permanent virtual connections (SVC or PVC) established between all user sites, yielding a fully meshed connectionless network. When SVCs are used, the connections are established and released according to the arrival and duration of connectionless sessions. This approach suffers from significant connection establishment overheads and delays, since packets must be queued until the VC to the destination is set-up. When PVCs are used, all connections are established at the same time and kept open permanently. The major drawback of the PVC approach is that it does not scale well. The number of PVC grows quadratically with the number of user sites, so does the routing table and the table lookup time. Furthermore both approaches suffer from poor utilization of resources since bandwidth is fragmented between all source destination pairs.
Chapter 5. Connectionless enterprise ATM network design

With the direct method, connectionless service is realized through the use of connectionless servers (CLSs) placed throughout the ATM network. Each user site simply forwards segmented connectionless packets to the nearest CLS, which is then responsible for making the routing decision and determining the next-hop CLS or the destination user site [VS94, CS94]. The CLSs are connected with PVCs in order to reduce connection setup delays, basically forming a connectionless overlay network. Since the overlay network does not necessarily include a VP for each CLS pair, the direct method reduces the number of PVCs with respect to the indirect method, which in turn lowers the scalability concerns. In addition it also simplifies VP routing table management and reduces bandwidth fragmentation. Hence in the following I consider only enterprise networks that employ the direct method.

In contrast to the previous chapter, this chapter is concerned with the design of a CLS overlay network on top an ATM architecture. In that context, an enterprise network is seen as a separate private VP subnetwork that coexists with other public VP subnetworks within the same ATM network. Furthermore this chapter considers network design from the enterprise (user) prospective, as opposed to the network provider's prospective which was considered previously. Hence the objective here is to design a CLS overlay network that satisfies all traffic demands and their QoS requirements while minimizing the enterprise network costs.

The rest of the chapter is organized as follows: the CLS overlay network operation and performance tradeoffs are discussed in the next section. The design problem is defined in section 5.2. It is formulated as a non-linear, mixed-integer programming problem. Because of its complexity, it is very difficult to develop exact optimization methods for obtaining of the global optimal solution. Therefore section 5.3 presents a heuristic algorithm, also based on simulated annealing, that produces suboptimal solutions with relative cost difference within only a few percent of the cost of the global optimum. The quality of the heuristic algorithm is discussed in section 5.4 as well as some experimental results. The chapter's summary is given in section 5.5.

5.1 Network architecture and architecture issues

As already mentioned, the interLAN networks considered in this chapter, rely on the direct method for exchange of connectionless data, i.e. on a set of non-ATM switching equipment (CLS), interconnected with PVCs forming a connectionless overlay network. Typically a user site\(^1\) will consist of several LANs or a MAN, connected through a interworking unit (IWU) to the ATM backbone (see figure 5.1). The IWU's only function is to segment or reassemble connectionless packets (datagrams) into/from ATM cells.

Because the IWU does not perform any routing functions, each user site needs at least one virtual connection to the nearest CLS attached to an ATM switch (see figure 5.1). A CLS can operate in one of the two forwarding modes: cut-through or reassemble mode. The cut-through mode relies on the use of AAL3/4 (see [Tel93b, CS94] for definitions).

\(^1\)The term user site denotes a corporate site that needs to exchange information with other geographically distant corporate sites.
5.1 Network architecture and architecture issues

Figure 5.1: An example of an enterprise CLS overlay network serving three corporate sites.

When the first cell of the segmented datagram arrives at the CLS, the CLS extracts the destination address from the ATM payload, determines the next-hop PVC to the destination, creates entries in the routing table for the corresponding VCI/VPI and multiplexing identifier (MID) input and output values, and immediately sends the ATM cell to the appropriate outgoing link. All subsequent cells belonging to the same datagram, can be efficiently routed by indexing the routing table with the MID value carried by each cell. The problems stem from the large overhead of the AAL3/4 protocol data unit and the very short time available for processing the cells. The reassemble mode relies on the use of AAL5 (see [Tel93b, CS94]). In this mode, the CLS reassembles the whole datagram before it forwards it to the next hop. The obvious problems with this mode of operation are the additional CLS queuing delay incurred at each hop and the complex buffer management. The tradeoffs between the two methods are discussed in more detail in [VS94, CFG+93]. Nevertheless for the design method discussed here both modes are applicable.

Thus when an user site wants to communicate (connectionless) with another distant user site, it does not attempt to establish a direct connection to the destination site, but instead it sends its datagrams to the closest CLS. The connection towards the CLS is determined by the ATM cell header VPI/VCI fields, regardless of the information in the datagram's destination field. The CLS then processes the datagram or the ATM cell depending on the forwarding mode, and forwards it to the next-hop CLS. So the datagram hops across the network until it reaches the CLS which delivers it to its destination site.

The efficiency of the CLS overlay network is influenced by several factors, like number and location of CLSs, overlay network topology, bandwidth allocation, destination address resolution etc. This chapter considers the first three factors.

Since it is unlikely that CLSs will be placed on top of every ATM switch, the number
and the placement of the CLSs has important performance consequences. In general, the fewer the CLSs nodes, the higher the load on each CLS and the load on each PVC in the overlay network. High CLS load is generally not desirable because the CLS must process the packets of all incoming links and can therefore easily become a performance bottleneck. High PVC load on the other hand, is beneficial since it concentrates the connectionless traffic to fewer links, thus potentially increasing the statistical multiplexing gain. Another important consideration is, of course, the cost of an CLS, which would drive towards networks with fewer CLSs.

Since the CLSs are not likely to be connected with a complete mesh of PVCs, the overlay network topology is also an important factor which determines the performance and the flexibility of the connectionless network. Because of the connection-oriented character of the underlying ATM network, virtually any overlay topology is feasible. Different topologies have been proposed (see [VS94] and references therein), ranging from arbitrary to centralized hierarchical topologies. Here I consider arbitrary topologies because in principle, they are the most general and in addition they also minimize the size of routing tables in the CLS nodes. Furthermore the network topology can also range from fully meshed (maximum connectivity) to a spanning tree (minimum connectivity). In general, high connectivity leads to traffic fragmentation and insufficient use of bandwidth. Low connectivity on the other hand, alleviates this problem but it also increases the CLS loads because the datagrams are forced to traverse more CLSs on average, before reaching their final destination.

Since connectionless enterprise networks should provide some degree of guaranteed service, bandwidth resources must be allocated to the PVCs interconnecting the CLSs. Naturally, as any other allocated resource the link bandwidth may be readjusted during the lifetime of the subnetwork, due to changes in the connectionless traffic flux. One approach is to let the CLSs dynamically change the bandwidth allocation on their outgoing links on a per burst or per connection basis. By monitoring the output buffers of the CLSs, bandwidth can be dynamically requested or released from/to the ATM network. This however does not seem to work very well in a CLS overlay networks, for several reasons. First it imposes extra load on the CLSs in a form of additional signaling traffic and procedures. Secondly, it adds additional bandwidth renegotiations delay at each CLS, which increases CLS congestion and furthermore it can prove worthless if the bandwidth renegotiations take longer than the time interval over which the traffic fluctuates. Finally the implementation of the dynamic bandwidth allocation is further complicated because of the potential hysteresis effects which may lead to instability [VS94, CFG+93]. Therefore in the rest of the chapter, it is assumed that the bandwidth allocation is quasi-static, i.e. the bandwidth allocation is static with respect to the BURST and CONNECTION level traffic changes, but it can be readjusted periodically at the PATH level, to reflect major shifts in traffic patterns².

Previous research efforts have focused on designing an optimal overlay topology, assigning CLS according to some heuristic performance considerations [Mon94, MFG94]. The solution algorithms are usually based on some special feature of the objective function,

²In depth analysis of the bandwidth allocation problem for various approaches, can be found in [CFG+93].
typically convexity. However, in reality the objective functions are neither convex nor concave, and often comprise integer and continuous variables. Therefore the design problem, which is defined in the next section, reflects all of the issues discussed above.

5.2 Network modeling and problem formulation

Before formally defining the design problem, I describe the underlying network model. The network model consists of a set of ATM nodes, CLSs and PVCs interconnecting them (see figure 5.2). The model does not include corporate user sites, because the first leg connections between the IWUs and the edge backbone nodes, have very little effect on the overlay network design. Instead it is assumed that the edge backbone nodes (see figure 5.1) are the sources of connectionless traffic, which in fact may originate from several distinct corporate sites. An ATM node which does not generate traffic, can be used as a transit node where CLS can be placed. It is assumed that all CLSs are of the same type and that they can be placed at every ATM node, including the edge nodes.

A CLS is connected to other CLSs through a PVC overlay network. The traffic sources on the other hand are connected to the closest CLS. It is assumed that only one PVC can be established between any pair of nodes and that all PVCs are bidirectional but not necessarily with the same bandwidth in both directions because of the asymmetric nature of the connectionless communications.

![Network model of the CLS overlay network used in the optimization.](image)

The design problem is described as follows: how many CLSs should be implemented, where should they be placed in the ATM network and how should they be connected,

---

3Note that an edge node can be both a traffic source and a CLS.
both in terms of topology and bandwidth allocation, in order to obtain the minimal cost overlay network and in the same time to be able to guarantee that the traffic will reach its destination with the requested QoS. The problem can be classified as a distributed network optimization problem where the network cost is minimized, subject to the offered traffic and the congestion measure.

The known parameters are: the coordinates of the ATM nodes, the average traffic demands and the costs associated with the links and the CLSs. The traffic between any pair of nodes is given by a traffic matrix $\Lambda = \{\lambda_{ij}\}$, where $\lambda_{ij}$ signifies the mean traffic from node $i$ to node $j$ in [kbits/s]. The associated costs are difficult to determine exactly, primarily due to the lack of CLS products and unwillingness of the operators to reveal high speed link prices. Therefore the employed cost function is based on some generic pricing rules and expectations of future developments. It consists of link cost part and CLS cost part. The link cost comprises a variable part, which is dependent on the link capacity and length, and a fixed part, which incorporates costs associated with the establishment and administration of the PVC. Furthermore it is assumed that only finite set of capacity values can be obtained from the ATM network making the link cost function discrete.4

Similarly the CLS cost comprises of a variable part, which depends on the maximal traffic flow that the CLS can handle, i.e. its capacity, and a fixed part, which expresses the cost of purchasing a CLS. Again it is assumed that CLSs come in only several capacities, thus the CLS part of the cost function is also discrete. All costs are expressed on per month basis.

The average queuing delay is often seen as an inappropriate measure of congestion in ATM networks, primarily because of the high link speeds involved.5 The buffer overflow probability (BOP) is considered as a more natural choice [RMV96, Onv94]. The problem with the BOP is that in general it is very difficult to determine it accurately and in general there are no closed form results available, except in a very few cases. However, the BOP is related to the mean link queue length, and the later is related to average queuing delay through the Little’s Law. Since the average queuing delay is much easier to calculate than the BOP, the analysis uses the average queuing delay as an indirect measure of the congestion in the CLS overlay network. Similar approach is also taken in [Mon94, MFG94].

The basic ingredients of the optimization model are: the set of nodes $N$, the set of CLSs capacities $Q$ and the set of link (PVC) capacities $C$. The optimization model also employs multiple commodities to distinguish between distinct origin destination (od) packet flows. Consequently two types of variables are used in the formulation: zero-one or binary decision variables, and continuous multicommodity flow variables. The binary variables are defined as:

$$x^c_{ij} = \begin{cases} 
1 & \text{if link of capacity } c \text{ is placed between nodes } i \text{ and } j, \\
0 & \text{otherwise.} 
\end{cases} \quad (5.1)$$

$$y^q_i = \begin{cases} 
1 & \text{if a CLS of capacity } q \text{ is attached to node } i, \\
0 & \text{if no CLS is present at location } i. 
\end{cases} \quad (5.2)$$

4 Figure 5.4 in section 5.3.1 gives an example of a sample link part of the cost function.

5 For instance on a 150 Mbit/s link, typical queuing delays will be in order of few tens of $\mu$s, whereas the propagation delays on a cross-country connection will be about few tens of ms.
The three dimensional matrix $X = \{x_{ij}^c\}$, determines the overlay network topology and the link capacities, whereas the two dimensional matrix $Y = \{y_i^q\}$, denotes the capacity and the location of the CLSs throughout the ATM network.

The continuous multicommodity flow variables are used to model the flow of connectionless traffic throughout the network. The variable $f_{ij}^{od}$, expresses the average flow in [bits/s], sent by source $o$ to destination $d$, on the link between sites $i$ and $j$. The traffic flow $f_{ij}$, is defined as the total traffic traversing link $ij$:

$$f_{ij} = \sum_{o \in \mathcal{O}} \sum_{d \in \mathcal{D}} f_{ij}^{od}.$$  \hspace{1cm} (5.3)

Based on the network model and the latest definitions the optimization problem is defined as:

**Given:** $\Lambda, \mathcal{N}, T_{\text{max}}, \{e_{ij}^c(l_{ij})\}$ and $\{d_i^q\}$

**Minimize:**

$$\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} \sum_{c \in \mathcal{C}} e_{ij}^c(l_{ij}) x_{ij}^c + \sum_{i \in \mathcal{N}} \sum_{q \in \mathcal{Q}} d_i^q y_i^q$$  \hspace{1cm} (5.4)

**Subject to:**

$$\sum_{j \in \mathcal{N}} f_{ji}^{od} - \sum_{j \in \mathcal{N}} f_{ij}^{od} = \alpha_i^{od} \hspace{1cm} \forall i, \forall o, \forall d \in \mathcal{N}$$  \hspace{1cm} (5.5)

$$\sum_{j \in \mathcal{N}} \sum_{c \in \mathcal{C}} c x_{ij}^c \leq \sum_{q \in \mathcal{Q}} q y_i^q \hspace{1cm} \forall i \in \mathcal{N}$$  \hspace{1cm} (5.6)

$$f_{ij} \leq \sum_{c \in \mathcal{C}} c x_{ij}^c \hspace{1cm} \forall i, \forall j \in \mathcal{N}$$  \hspace{1cm} (5.7)

$$\sum_{c \in \mathcal{C}} x_{ij}^c \leq 1 \hspace{1cm} \forall i, \forall j \in \mathcal{N}$$  \hspace{1cm} (5.8)

$$\sum_{q \in \mathcal{Q}} y_i^q \leq 1 \hspace{1cm} \forall i \in \mathcal{N}$$  \hspace{1cm} (5.9)

$$T \leq T_{\text{max}}$$  \hspace{1cm} (5.10)

The objective function (5.4) is a measure of the cost of all PVCs and CLSs in the overlay network over certain time interval. The cost of connecting CLSs located at nodes $i$ and $j$ with a PVC of capacity $c$, denoted by $e_{ij}^c(l_{ij})$, is dependent on the distance between the nodes $i$ and $j$, $l_{ij}$, due to operator's tariffs. The cost of CLS of capacity $q$ at node $i$, is denoted by $d_i^q$ and is due to equipment manufacturer prices.

Constraints (5.5) are the usual flow conservation equations. Basically they make certain that for an $od$ pair, the flow entering a CLS must equal the flow exiting the CLS, unless the node is the actual destination or origin. Thus $\alpha_i^{od}$ is defined as:

$$\alpha_i^{od} = \begin{cases} 
  a_i^{od} & \text{if } i = d \text{ (demand)}, \\
  -a_i^{od} & \text{if } i = o \text{ (supply)}, \\
  0 & \text{otherwise},
\end{cases}$$  \hspace{1cm} (5.11)
where $a^{od}$ specifies the mean traffic flow from node $o$ to node $d$.

Constraints (5.6) ensure that the CLS has enough capacity to support all connections at node $i$. The total capacity of all links connected to node $i$ is constrained to be less than or equal to the CLS processing speed $q$, in [bits/s]. Constraints (5.7) limit the flow on each link to the link capacity, i.e. the total flow on each link, $f_{ij}$, may never exceed the link capacity, $c$. Constraints (5.8) ensure that only a single link connects nodes $i$ and $j$, whereas constraints (5.9) ensure that only one CLS is placed at each node $i$. Constraint (5.10) controls the level of congestion within the overlay network. It ensures that the average network wide delay does not exceed a certain value, $T_{\text{max}}$.

The delay expression will of course depend on whether reassembly or cut-through mode is used in the CLS. The analysis is further complicated by the segmentation of the datagrams into ATM cells, which results in interleaving several datagrams on the same link. The delay clearly depends on the assumptions used, and is generally not available in closed form [Hei95]. Therefore the numerical examples in section 5.4, use the following approximation [GK77]. Assuming Poissonian arrivals, exponential packet lengths, infinite buffers and independence between interarrival transmission times on subsequent links (the famous Kleinrock assumption), i.e. analyzing each link as an independent M/M/1 queue, the average network wide delay can be found as:

$$T = \frac{1}{\gamma} \sum_{i \in N} \sum_{j \in N} \frac{f_{ij}}{C_{ij} - f_{ij}},$$

(5.12)

where $\gamma$ is a constant related to the total input rate in the network and $C_{ij} = \sum_{c \in C} c x_{ij}^c$ denotes the capacity of the link between nodes $i$ and $j$.

Note that equation (5.12) introduces non linearity in the optimization model since the average network delay is a non linear function of the flows $f_{ij}$. Although the approximation is rather crude and there are better approximations available [Sch87, GK77], it is sufficient for our purpose and furthermore the solution procedure described in the next section can be easily extended to more complicated and accurate expressions.

The described problem, is highly complex non linear mixed integer program, known to be NP-complete. There are no generic mathematical programming methods available for solving it. Hence the next section discusses a heuristic technique based on the simulated annealing algorithm.

### 5.3 Solution procedure

The complexity of the optimization problem described in the previous section, stems from the fact that in general the objective function is neither convex nor concave, and from the presence of both integer and continuous valued variables. In addition, both the objective function and the performance constraint (5.10), are subject to change, the former due to varying tariffs and the later due to the possibility of including more accurate performance measures. In such cases, conventional mathematical programming and network flow techniques are either not applicable or yielding local minima only. On the other hand, heuristic combinatorial techniques, like the simulated annealing described in
5.3 Solution procedure

chapter 4, are typically independent of the mathematical form of the objective function and the functional constraints, as well as from the type of variables involved. The solution procedure used here is based on the simulated annealing algorithm, which makes it inherently flexible and adaptable to changes.

The problem belongs to the class of general network design problems [GK77, Ker93]. The objective is to determine the location and the capacity of the CLSs and the interconnecting topology, at a minimum cost, satisfying certain level of performance. This however is closely related to the decisions on how to route traffic through the network and what capacities to assign to the network links. Clearly the general network design problem is very complex, and even its subproblems, like the capacity and flow assignment problem, are known to be difficult. Therefore the solution approach taken here is to subdivide the general problem into two related subproblems: CLS placement and link topology optimization.

The optimization procedure incorporates two nested annealing loops, an outer and an inner loop, corresponding to each subproblem respectively. The inner loop optimizes the connection overlay network for a given positioning of the CLSs, and the outer loop optimizes the number and the location of the CLSs. The flow chart of the optimization procedure is given on figure 5.3.

![Flow Chart](image)

Figure 5.3: A flow chart of the optimization procedure.

The optimization process begins by initializing all of the necessary parameters, finding an initial feasible solution and calculating its cost. Then the outer loop is entered, where a new CLS configuration is generated. For each iteration of the outer loop, the optimal connection overlay network (link placement and dimensioning) is found by executing the
whole inner annealing loop. At each iteration of the inner loop, the algorithm evaluates the link flows and delays, and determines the link and CLS capacities. When the inner loop is finished, the procedures returns to the outer loop, where the new network is compared with the current optimum. If the new solution is cheaper according to the annealing rule, then it is accepted as the new best solution. Otherwise the optimization continues by changing the previous solution following the outer loop annealing schedule. When subsequent configurations, repeatedly do not deliver any significant cost reductions, the outer loop exits and the current optimum becomes the final solution of the optimization process.

5.3.1 Algorithm implementation

Simulated annealing is a general optimization algorithm, applicable to a wide variety of problems. As discussed in the previous chapter, in order to apply the algorithm to the problem at hand, one must define the system configuration description (vector space), the generator of random configuration changes (neighborhoods), the objective function and the cooling schedule. Since there are two nested annealing loops in the optimization algorithm, there are also two implementations of the algorithm.

In the inner loop, the configuration is determined by the location and the capacity of the links connecting the CLSs, as well as the CLSs and the ATM nodes. Changes of the configuration, performed in the Shuffle connections block in figure 5.3, are achieved by randomly executing one of the following three operations:

- **add a link**, adds a link between randomly chosen CLS nodes,
- **remove a link**, removes the link between randomly chosen CLS nodes,
- **move a link**, removes a link between randomly chosen CLS nodes and then adds it back between other two randomly chosen CLS nodes.

Thus the subsequent configurations differ in only one link. Note though that this is true only for the topology, the capacity of the links can be very different in the two configurations.

The objective function has the same form as the objective function of the overall problem given with equation (5.4). It consists of two terms, the link cost term and the CLS cost term. The link cost depends on the link capacity and its length. A sample link cost function is depicted in figure 5.4.a. It shows the familiar economy of scale effect for the link costs.

The CLS cost term incorporates a fixed offset and variable part which depends on the capacity of the CLS. Since the capacity of the PVCs change in the inner loop, so does the CLSs capacity. Hence the CLS cost term is also significant in the inner annealing loop. A sample CLS cost function is depicted in figure 5.4.b. It assumes that CLS cost increases stepwise with the increase of the capacity. Note that other cost functions can also be used since the annealing algorithm is independent of the mathematical form of the objective function.

The cooling schedule is the same as in chapter 4 (see subsection 4.3.2), based on the algorithm proposed in [JAMS89, JAMS91]. The relevant input cooling parameters of the
5.3 Solution procedure

![Graphs showing link and CLS cost functions.]

Figure 5.4: A sample cost function of the optimization procedure: a) link cost function, b) CLS cost function.

The algorithm are: the INNITTEMP, SIZEFACTOR, $T_{\text{decay}}$, CUTOFF, MINPERCENT and FROZEN. After elaborate experimentation, the values were set to the following values: 10, 8, 0.81, 0.4, 2% and 3, respectively. These values will also be used in the experiments discussed in section 5.4.

In the outer loop, the system configuration is determined by the number, location and the capacity of the CLSs. Similarly as in the inner loop, the configuration changes, performed in the Shuffle Servers block, are generated by randomly executing one of the following three operations:

- **add a CLS**, adds a CLS at randomly chosen ATM node,
- **remove a CLS**, removes a CLS from a randomly chosen ATM node,
- **move a CLS**, removes a CLS from randomly chosen ATM node $j$ and then adds it back at a randomly chosen ATM node different than $j$.

Again the subsequent configurations differ only in one CLS. The capacities of the CLS are of course different in the both configurations.

The objective function is the same as in the inner loop. Both the link cost and the CLS cost terms are relevant because both of them vary in the course of the algorithm. The cooling schedule has the same structure as in the inner loop. The input parameters, however are slightly different: INNITTEMP = 10, SIZEFACTOR = 4, $T_{\text{decay}} = 0.9$, CUTOFF = 0.4, MINPERCENT = 2% and FROZEN = 3.

### 5.3.2 Routing and Link dimensioning

The last block from figure 5.3 that needs to be defined is the **Routing and link dimensioning**. At each iteration of the inner annealing loop, it is necessary to determine the traffic flows $f_{ij}$ and the link capacities $C_{ij}$ that satisfy the delay constraint (5.10), i.e. one should solve the so-called capacity flow assignment (CFA) problem [GK77, Ker93]. The CFA problem has been extensively researched. If one assumes linear costs and an $M/M/1$ queuing network, it is possible to express the optimal capacities $C_{ij}$ in terms of the flows...
so that the problem is redefined as an routing problem and subsequently solved with
the Flow Deviation (FD) algorithm (see e.g. [GK77, Ker93, FGK73]). In case of concave
costs, the problem can be solved by approximating the cost function with a continuous
power law function and then applying the FD algorithm again [GK77].

However the costs in real networks are discrete, which significantly complicates the
exact analysis. The solution algorithms are mainly based on two heuristic approaches:

(i) solve, iteratively a routing problem for fixed capacities, followed by a discrete ca-
pacity assignment problem with fixed flows, until a local minimum is obtained;

(ii) interpolate discrete costs with continuous, concave costs and solve the corresponding
concave CFA problem.

The solution procedure in this chapter follows the first approach. The employed algorithm
is an adapted version of the Minimum Link Assignment algorithm proposed in [GK77].
It iterates through a routing and a capacity assignment steps until subsequent iterations
stop producing significant improvements. First assuming infinite link capacities, find
the shortest path for each od pair. The shortest path criterion is based on the linear
combination of the minimum hop count and the distance between the ATM nodes. If
multiple shortest paths are available, the traffic is routed along the least loaded path.
The link flows $f_{ij}$ are determined, after processing all od pairs. Then the link capacity
is obtained from the link flow and the delay constraint. This is done by assigning a
parameter $\alpha \in [0, 1)$, $\alpha \leq \frac{T_{\text{max}}}{a_1 + T_{\text{max}}}$ such that the capacity of each link is obtained as the
smallest discrete capacity greater than $f_{ij}/\alpha$, i.e.:

$$C_{ij} = \min_{c \in C} \{c \mid c \geq \frac{f_{ij}}{\alpha} = f_{ij}(1 + \frac{a_1}{T_{\text{max}}})\},$$

(5.13)

where $a_1$ is a positive constant depending on the total input rate. Thus by varying $T_{\text{max}}$
one can obtain networks with different levels of residual bandwidth.

The main advantage of this algorithm is that it is computationally simple and thus very
suitable for application in CPU intensive algorithms like the simulated annealing.

5.4 Experimental results

In order to analyze the proposed optimization procedure, we have developed an proto-
type software tool called, Bnet, which incorporates the design procedure outlined in the
previous section. It comprises three parts: a minimum-cost assignment algorithm already
discussed, a routing algorithm which finds possible routes for the network traffic and a
user-interface which helps the user to study the effects of the different input parameters
on the resulting network configuration (for more details on Bnet, see [SdJvL95, SS95]).

This section describes two experiments, which illustrate the application of the des-
dign tool. The first experiment investigates the quality of the results obtained from the
optimization. The second experiment investigates the impact of varying the CLS costs
(relative to the link costs) on the optimal solution.
5.4 Experimental results

5.4.1 Quality of the optimization procedure

Typically the quality of the heuristic algorithms is determined the duality gap between the primal feasible solution and a lower bound. Lower bounds are usually obtained by finding the minimum spanning tree and approximating the discrete costs with linear or concave functions. Linear bounds are often too loose because of the economy of scale effect which is present in the cost function, whereas concave bounds are very difficult to obtain because the dependence between the costs and traffic flows can not be expressed in closed form. A lower bound can be obtained by setting the capacities of all links in the minimum spanning tree to the minimal capacity. Unfortunately this bound is tight only if the network traffic is extremely low. For realistic network traffic levels, typical optimal solutions will be several times as large as the lower bound.

Since standard bounding techniques do not yield satisfactory results, I use the following approach. First the cost distribution of an arbitrary feasible network is estimated by generating a large number of random networks and then the cost of the optimal solutions produced by the optimization procedure are compared relative to the general cost distribution.

In order to estimate the general cost distribution another tool was developed, called the RandomNet [BK96]. Given the traffic demands, the tool can generate any number of distinct feasible networks and calculates their costs. It first generates a network with arbitrary number and location of CLS, and random topology. Then the traffic is routed randomly assuming infinite capacities, and if all traffic can be routed, the link capacities are reduced to the minimal values that still satisfy the delay constraint. Finally the cost of the new network is calculated. The whole procedure is repeated until the desired number of sample networks is generated. The tool keeps track of all generated networks to prevent counting the same network twice. Note that in the generation process there is no attempt to optimize CLS placement, link topology or network routing. The link capacity is minimized in order to avoid unnecessarily biasing the cost distribution with networks with disproportional amount of resources.

The experiment was performed on the network given on figure 5.5. The network represents SURFNet, the Dutch university network. It consists of thirteen nodes spread around Netherlands. The traffic matrix used in the experiment, see table 5.1, is based on traffic measurements obtained from the current operational network [Rei96]. The cost function used is the same as the one defined in section 5.2 (see figure 5.4).

The histogram of the cost of 2.099.944 arbitrary feasible networks, is depicted in figure 5.6. As expected it shows the familiar characteristics of the Gaussian distribution. Hence the quality of the annealing solution can be determined by using the estimate:

\[ \Pr\{x > C(x^*)\} = \frac{1}{\sqrt{2\pi} \sigma_x} \int_{C(x^*)}^{\infty} e^{-\frac{(x-x)^2}{2\sigma_x^2}} dx, \]  

(5.14)

where \( \bar{x} \) and \( \sigma_x \) are the mean and the standard deviation of a feasible topology, respectively, and \( C(x^*) \) is the cost of the solution obtained by the optimization procedure of section 5.3.1. Both \( \bar{x} \) and \( \sigma_x \) can be obtained from the histogram in figure 5.6. The rational behind the estimate is that the general network distribution is approximated with an

\[ \text{An arbitrary feasible network is a network that satisfies all constraints of the optimization problem (equations (5.5) through (5.10)).} \]
Figure 5.5: A screenshot of SURFNet, showing the result of the optimization.

<table>
<thead>
<tr>
<th>Node</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>58.8</td>
<td></td>
<td>8.0</td>
<td>2.0</td>
<td>20</td>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>11.0</td>
</tr>
<tr>
<td>2</td>
<td>32.0</td>
<td>-</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5.0</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3.0</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>11.0</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10.0</td>
</tr>
<tr>
<td>6</td>
<td>41.0</td>
<td></td>
<td></td>
<td>30.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>10.0</td>
<td></td>
<td></td>
<td></td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>20.0</td>
<td></td>
<td></td>
<td>10.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>7.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>30.0</td>
</tr>
<tr>
<td>10</td>
<td>25.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9.0</td>
<td></td>
<td></td>
<td>20.0</td>
</tr>
<tr>
<td>11</td>
<td>40.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>24.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>12.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>

Table 5.1: SURFNet traffic demand matrix in [Mbits/s], adapted IP traffic measurements, August 1993. Row number denotes the source node, column number denotes the destination node.

Gaussian distribution with appropriate mean and standard deviation, which allows us to estimate the percentage of possible network configurations that are more expensive than the annealing solution.

The results are presented in Table 5.2. The cost of the annealing solution, \( C(x^*) \), is
5.4 Experimental results

Figure 5.6: Histogram of the cost of an arbitrary connectionless overlay based on SURFNet network parameters.

obtained as the average cost of 30 runs of the optimization algorithm\(^7\). The results show that the solutions obtained by the optimization algorithm, are better than 99.98% of all feasible solutions. The average cost of an arbitrary network is \$15,865.59, whereas the mean annealing solution costs only \$12,197.66. This clearly shows the superiority of the optimization procedure compared to random search algorithms. The lowest cost found by the annealing algorithm was \$11,840 and the corresponding network is given on figure 5.5. More details about the numerical results as well as about the RandomNet tool can be found in [BK96].

| \(|N|\) | \(C(x^*)\) | \(\bar{x}\) | \(\sigma_x\) | \(Pr\{x > C(x^*)\}\) |
|---|---|---|---|---|
| 13 | 12,119 | 15,865.59 | 1026.98 | 0.999875 |

Table 5.2: Results of the first experiment

5.4.2 CLS costs vs. link costs

The second experiment is inspired by the absence of the CLS placement considerations in the CLS overlay design literature. Namely, most of the available algorithms do not incorporate the CLS placement phase, and the number and location of the CLS servers are usually determined by examining some performance arguments [Mon94, MFG94]. Therefore this section considers the impact of the relation between the CLS costs and link costs on the structure of optimal CLS overlay network.

\(^7\)Since simulated annealing is a stochastic algorithm, different runs can produce different solution, hence the averaging of \(C(x^*)\).
In principle, for a given traffic matrix, the structure of the overlay network will mainly depend on the values of the objective function parameters. Since I want to investigate the impact of the relation between the CLS costs and link costs, I keep the link cost parameters constant and vary only the CLS cost function parameters. Recall from section 5.2 that the sample CLS cost function was defined as an initial offset and a discrete step function with a linearly increasing trend. Hence the goal is to investigate how does the optimal CLS overlay network change as a function of the offset and the trend parameters.

The sample network used in this set of experiments is the backbone of UNINETT, a network that connects the universities, colleges and other research institutions in Norway. The network together with a list of node names, is depicted in figure 5.7.

<table>
<thead>
<tr>
<th>Nr</th>
<th>Site</th>
<th>Nr</th>
<th>Site</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Alta</td>
<td>9</td>
<td>Stavanger</td>
</tr>
<tr>
<td>2</td>
<td>Bergen</td>
<td>10</td>
<td>Lillehammer</td>
</tr>
<tr>
<td>3</td>
<td>Molde</td>
<td>11</td>
<td>Mo</td>
</tr>
<tr>
<td>4</td>
<td>Elverum</td>
<td>12</td>
<td>Narvik</td>
</tr>
<tr>
<td>5</td>
<td>Gjøvik</td>
<td>13</td>
<td>Oslo</td>
</tr>
<tr>
<td>6</td>
<td>Grimstad</td>
<td>14</td>
<td>Grondheim</td>
</tr>
<tr>
<td>7</td>
<td>Halden</td>
<td>15</td>
<td>Tromsø</td>
</tr>
<tr>
<td>8</td>
<td>Harstad</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.7: Network architecture of UNINETT, corresponding to the optimal solution with sequence number 3 of table 5.4. The table on the left lists the node names.

Table 5.3 presents the matrix of traffic demands used in the experiments. The matrix was obtained from measurement data available at the provider’s web site (www.uninett.no). It contains the average weekly busy hour traffic intensities, which were chosen as the appropriate traffic statistics for dimensioning of the network. More details on the extraction of the traffic statistics can be found in [HvEG97].

The cost function is the same as in the previous subsection, only now the parameters of the CLS cost term are varied. Since the CLS cost term contains two parameters, two sequences of experiments were performed, one investigating the impact of the cost offset and another one investigating the impact of the linear trend.

The results are presented in tables 5.4 and 5.5. Table 5.4 shows the results for varying the cost offset. The trend is set to zero, i.e. it is assumed that the CLS cost is independent of the server capacity. The results show that as the offset increases, which means that the CLSs are becoming more expensive, the number of servers in the optimal solution decreases and vice versa. For low values of the offset parameter, the number of CLSs in the optimal network is high. For values around 1000, the number of CLSs oscillate
<table>
<thead>
<tr>
<th>#</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>452</td>
<td>142</td>
<td>136</td>
<td>144</td>
<td>218</td>
<td>803</td>
<td>49</td>
<td>281</td>
<td>70</td>
<td>76</td>
<td>133</td>
<td>4292</td>
<td>452</td>
<td>452</td>
</tr>
<tr>
<td>2</td>
<td>1830</td>
<td>-</td>
<td>3946</td>
<td>3787</td>
<td>4012</td>
<td>6069</td>
<td>22315</td>
<td>1362</td>
<td>7810</td>
<td>1948</td>
<td>2109</td>
<td>3692</td>
<td>90088</td>
<td>10566</td>
<td>10566</td>
</tr>
<tr>
<td>3</td>
<td>68</td>
<td>470</td>
<td>-</td>
<td>140</td>
<td>148</td>
<td>224</td>
<td>824</td>
<td>50</td>
<td>7288</td>
<td>72</td>
<td>78</td>
<td>136</td>
<td>4404</td>
<td>528</td>
<td>470</td>
</tr>
<tr>
<td>4</td>
<td>163</td>
<td>1134</td>
<td>352</td>
<td>-</td>
<td>358</td>
<td>542</td>
<td>1993</td>
<td>122</td>
<td>697</td>
<td>174</td>
<td>188</td>
<td>7330</td>
<td>710778</td>
<td>1134</td>
<td>1134</td>
</tr>
<tr>
<td>5</td>
<td>212</td>
<td>1475</td>
<td>458</td>
<td>439</td>
<td>-</td>
<td>704</td>
<td>2587</td>
<td>158</td>
<td>906</td>
<td>226</td>
<td>244</td>
<td>428</td>
<td>14013</td>
<td>1475</td>
<td>1475</td>
</tr>
<tr>
<td>6</td>
<td>322</td>
<td>2265</td>
<td>694</td>
<td>666</td>
<td>705</td>
<td>-</td>
<td>3923</td>
<td>239</td>
<td>1373</td>
<td>342</td>
<td>371</td>
<td>649</td>
<td>21803</td>
<td>2548</td>
<td>1699</td>
</tr>
<tr>
<td>7</td>
<td>1317</td>
<td>10214</td>
<td>2841</td>
<td>2726</td>
<td>2889</td>
<td>4370</td>
<td>-</td>
<td>980</td>
<td>5623</td>
<td>1403</td>
<td>1518</td>
<td>2658</td>
<td>98310</td>
<td>8937</td>
<td>10214</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>70</td>
<td>22</td>
<td>21</td>
<td>23</td>
<td>734</td>
<td>125</td>
<td>-</td>
<td>44</td>
<td>11</td>
<td>12</td>
<td>21</td>
<td>658</td>
<td>79</td>
<td>70</td>
</tr>
<tr>
<td>9</td>
<td>157</td>
<td>980</td>
<td>339</td>
<td>326</td>
<td>345</td>
<td>522</td>
<td>1920</td>
<td>117</td>
<td>-</td>
<td>168</td>
<td>181</td>
<td>318</td>
<td>10645</td>
<td>11261</td>
<td>1121</td>
</tr>
<tr>
<td>10</td>
<td>15</td>
<td>106</td>
<td>33</td>
<td>32</td>
<td>34</td>
<td>51</td>
<td>188</td>
<td>11</td>
<td>66</td>
<td>-</td>
<td>18</td>
<td>31</td>
<td>1004</td>
<td>106</td>
<td>106</td>
</tr>
<tr>
<td>11</td>
<td>27</td>
<td>182</td>
<td>57</td>
<td>55</td>
<td>58</td>
<td>88</td>
<td>323</td>
<td>20</td>
<td>113</td>
<td>28</td>
<td>-</td>
<td>54</td>
<td>1685</td>
<td>228</td>
<td>182</td>
</tr>
<tr>
<td>12</td>
<td>120</td>
<td>831</td>
<td>258</td>
<td>248</td>
<td>263</td>
<td>397</td>
<td>1461</td>
<td>89</td>
<td>511</td>
<td>128</td>
<td>138</td>
<td>-</td>
<td>7894</td>
<td>831</td>
<td>831</td>
</tr>
<tr>
<td>13</td>
<td>4493</td>
<td>138951</td>
<td>9691</td>
<td>9299</td>
<td>9854</td>
<td>14905</td>
<td>54801</td>
<td>3344</td>
<td>19180</td>
<td>4785</td>
<td>5179</td>
<td>9068</td>
<td>-</td>
<td>97265</td>
<td>41685</td>
</tr>
<tr>
<td>14</td>
<td>2008</td>
<td>133767</td>
<td>4332</td>
<td>4157</td>
<td>4404</td>
<td>6662</td>
<td>24495</td>
<td>1495</td>
<td>8573</td>
<td>2139</td>
<td>2315</td>
<td>4053</td>
<td>110822</td>
<td>-</td>
<td>13078</td>
</tr>
<tr>
<td>15</td>
<td>758</td>
<td>4341</td>
<td>1635</td>
<td>1568</td>
<td>1662</td>
<td>2514</td>
<td>9243</td>
<td>564</td>
<td>3235</td>
<td>807</td>
<td>873</td>
<td>1529</td>
<td>34730</td>
<td>4341</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 5.3: Average weekly busy hour traffic intensities in [kbit/s]. The row number indicates the source node and the column number the destination node.
between high and low values, which is characteristic of the transient region. For values higher than 2000 the number of CLSs stabilizes again at low values.

<table>
<thead>
<tr>
<th>Seq. #</th>
<th>offset</th>
<th>trend</th>
<th># of CLSs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>150</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>3000</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>5000</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>7000</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 5.4: Number of CLSs in the optimal overlay network for varying offset.

<table>
<thead>
<tr>
<th>Seq. #</th>
<th>offset</th>
<th>trend</th>
<th># of CLSs</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>150</td>
<td>0.2</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>150</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>150</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>150</td>
<td>10</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 5.5: Number of CLSs in the optimal overlay network for varying trend.

Why such a behavior? Recall that higher number of CLSs generally increases CLS costs, but on the other hand it could reduce bandwidth fragmentation and link length, and therefore decrease the total link costs. Smaller offset means that placing a new CLS becomes cheaper, and in general the total CLS costs are becoming lower relative to the total link costs. At a certain point the CLS price is such that placing more CLSs becomes cost effective because the increase in the CLSs costs is covered by the link cost savings due to higher economies of scale and shorter links that can be achieved because of the concentration of network traffic. Hence the number of CLSs in the optimal solution increases.

Table 5.5 shows the results for a trend varying between 0 and 10 and a fixed offset of 150. Again a similar behavior is noticed as in the previous case. As the trend increases the number of servers in the optimal solution decreases and vice versa. Such behavior can again be explained in line of the CLS costs vs. link costs tradeoff. Note that large trend implies that even a CLS with only one unit of capacity becomes prohibitively expensive. Therefore for high trends the link cost savings can never cover the increase in costs for purchasing multiple CLSs. Conversely for low trends the CLS cost tends to be insensitive to CLS capacity and hence at certain point installing more CLSs becomes economically viable.

Thus the conclusion of this subsection is that the CLS costs-link costs ratio, has very significant impact on the CLS overlay network structure, that must be taken into account in the design process. Hence heuristic considerations of the location and capacity of the CLSs outside the scope of the optimization ([Mon94, MFG94]) can lead to serious inaccuracies in the final design.

## 5.5 Conclusion

This chapter discusses the design of connectionless overlay network on top of an ATM network, as proposed by the ITU-T [Tel93b]. It investigates the impact of the number and the location of the CLSs, the overlay network topology and the bandwidth allocation on the overall network efficiency. The design problem is formulated as an non linear mixed-integer programming problem. The objective is to minimize link and CLS costs, subject to flow, capacity and delay constraints.
5.5 Conclusion

Because of the problem complexity, the solution procedure is based on the simulated annealing algorithm (see section 5.3.1). The procedure can be easily adapted to various types of objective functions, and functional constrains, mainly due to the inherent wide applicability of the annealing algorithm.

The quality of the solution procedure is investigated in subsection 5.4.1. It is shown that the procedure performs very good and obtains solutions within 0.01% of the optimum. Furthermore, it was also shown that the CLS-to-link cost ratio have very significant ramifications for the structure of the optimal network, and therefore it must be taken into account when designing a connectionless enterprise ATM overlay network.
Chapter 6

Conclusions

The main subjects of this thesis are the performance analysis and the design of ATM networks. The ATM network is envisioned as the common network platform that integrates data, video, voice, multimedia and any conceivable future service, while using its resources efficiently. The efficient use of the network resources is of paramount importance for both network providers and users, because of the explosive traffic growth that we are experiencing today and the billions of dollars that are spent annually on increasing the network capacity. However, supporting such a heterogeneous mixture of services also implies poor resource utilization because of the wide range of QoS requirements that the network needs to provide. Therefore the central question in this thesis is how to design an ATM network that efficiently shares its resources between all users, who have varied and often conflicting QoS requirements.

The thesis approach is based on stochastic modeling. The ATM network is modeled as a queuing network, where the arrival traffic streams are represented with "appropriate" traffic models and the network nodes are represented as queues connected through constant rate servers, corresponding to the ATM links. In order to determine the most appropriate traffic control mechanisms, the network traffic is divided into four traffic classes, LX, LD, LL and HX, according to its QoS requirements (loss and delay).

The analysis follows a multi-level approach: the single-link analysis considers an ATM link in isolation, whereas the network analysis considers the ATM network as a whole. The single-link analysis focuses on the LD and LL classes and it analyzes the potential performance improvements that can be achieved by applying a novel priority queuing mechanism. It also examines the validity of the traditional modeling and analysis techniques in the light of the newly discovered self-similar properties of the network traffic. The network analysis considers a new design framework that allows for efficient integration of all services on the same physical platform, at a minimal cost. Additionally it also considers the design of enterprise ATM overlay networks and the significance of various design variables for the overall design process.

The next section discusses the claims that follow from the research results described in the thesis. The last section discusses the future research topics that result from the limitations of the presented results, and the related research issues that are outside the scope of the thesis.
6.1 Claims

This section describes the main contributions of the thesis. The contributions are presented according to the chapter that treats the respective research issue.

Chapter 2 forms the basis for the ensuing chapters. It provides a classification framework for the network traffic according to the required level of service quality, a formal definition of self-similarity and a survey of traffic models according to the activity level of interest. The main conclusions of Chapter 2 are:

A.1 The classification framework distinguishes between four traffic classes, an LX, an LD, an LL and an HX class. Since the LX traffic class offers very limited possibilities for achieving any substantial efficiency gains, and the HX class is assigned the left-over bandwidth, the most promising classes, from a higher efficiency point of view, are the LD and the LL traffic class.

A.2 Chapter 2 also considers various traffic models at the CONNECTION, BURST and CELL level. The traffic behavior at the BURST time scale is such, that each traffic class requires a different traffic model.

A.3 Because of the presence of self-similarity in virtually all BURST level measured traces, the chapter also distinguishes between traditional and self-similar BURST level traffic models. The validation analysis of the presented models against the actual traffic measurements, has shown that the traditional models provide reasonably close representation of the actual traffic, whereas the self-similar models are less accurate. In particular the traditional models represent the SRD traffic properties well, but give rather poor representation of the LRD properties. Exactly the opposite holds for the self-similar models.

The main contribution of this chapter is the validation of the BURST level traffic models against the actual traffic measurements.

Chapter 3 is concerned with the performance analysis of a single ATM link. In particular, it analyzes traffic control mechanisms that allow higher resource utilization levels. The analysis incorporates both the CELL and the BURST time scales. At the CELL level, it examines the performance of the LDOLL queue, a novel priority queueing mechanism that multiplexes the LD and the LL traffic classes onto a single link, according to their respective QoS requirements. Additionally, the chapter also explores the possibilities for extending the LDOLL principle to an ATM buffer with multiple outlets that share the same buffer resources. At the BURST level, it examines the performance improvements of the LDOLL queue under more realistic circumstances and the impact of the self-similarity on the ATM buffer performance.

The conclusions of Chapter 3 include:

B.1 At the CELL level, the single-outlet LDOLL queue attains substantial performance gains with respect to the standard FIFO queuing, allowing higher link utilization levels.
B.2 The *myopic threshold* policies, although not optimal, are still the best practical policies for operating the multiple-outlet LDOLL queue, mainly due to their simple structure and a performance that is very close to the performance of the optimal policies. The multiple-outlet LDOLL queue, governed by the *myopic threshold* policies, attains significantly better performance than the shared FIFO and the separate LDOLL queuing strategies.

B.3 At the BURST level, the performance gains of the LDOLL queue under more realistic conditions (BURST level traffic variability and medium-size buffers) are generally lower than at the CELL level, but still significant (in the presented example the LDOLL queue could potentially provide a 15% higher utilization than the FIFO queue). Furthermore the gains become more pronounced for larger buffer sizes.

B.4 At the BURST level, for medium-size buffers, the self-similar traffic properties have only a minor effect on the buffer performance. As a consequence, the applicability of the traditional modeling and analysis techniques remains strong, although an additional validation effort is advisable.

The main contributions of Chapter 3 are the generalization of the LDOLL queuing strategy to multiple outlets and the BURST scale analysis results.

Chapter 4 is concerned with the design of ATM networks, in particular with the impact of the ATM multi-layer structure on the network efficiency. It focuses on designing the optimal configuration of the VP subnetworks on top of a given physical infrastructure, which minimizes network costs for given traffic demands and QoS requirements. Since the VP subnetwork configuration determines how the network resources are shared between the traffic service classes, determining the optimal configuration is crucial for achieving higher resource utilization levels.

The design problem is very complex and incorporates several subproblems. Previous research has mainly considered separate subproblems or specific resource-sharing strategies (typically complete sharing). The main contributions of Chapter 4 are:

C.1 A flexible *design framework* that captures the relevant design trade-offs, incorporates all related subproblems and can be easily extended to include additional functionality, like priority queuing, for instance. The framework is based on a heuristic procedure that iteratively solves a sequence of related subproblems.

C.2 BANDIT – a flexible and easy-to-use software tool that performs ATM network design by executing the *design framework*. Consequently it enables the user to examine the quality of the design framework and an arbitrary subproblem-solution algorithm.

The preliminary experimental results illustrate the quality and the applicability of the design framework, as well as the practicality of the software tool.

Chapter 5 is concerned with the design of a connectionless overlay network on top of an ATM network, as proposed by the ITU-T [Tel93b]. It considers the impact of the placement of the CLSs throughout the network, the overlay network topology and the
bandwidth allocation on the overlay network efficiency. Previous research efforts have focused on designing the overlay network based solely on the network topology and bandwidth allocation aspects, while the CLSs were usually assigned outside the main design process according to some heuristic arguments. Because of the complexity of the combined problem, the solution procedure incorporates two nested annealing loops, one for the topology and one for the CLS optimization, respectively. Its quality has been examined by employing a software tool called \textit{B-net}, which implements the solution procedure.

The main conclusions of Chapter 5 are:

D.1 The solution procedure produces results that are better than 99.98\% of all feasible solutions.

D.2 The relative CLS costs have very significant impact on the structure of the optimal overlay network, and therefore the CLS placement should not be determined outside the scope of the main design process.

The contributions of Chapter 5 include the solution procedure, the design tool and the relevance of the CLS placement result for the design process.

6.2 Recommendations for future research

This section discusses directions for future research. Future research efforts should be directed towards solving the limitations of the presented results as well as towards related research issues that are outside the scope of this thesis, but can be seen as a natural continuation of the principles and the ideas described in it. Future research is required in four areas: traffic modeling, \textsc{burst} level analysis, \textsc{connection} and \textsc{path} level analysis of the \textsc{ldoll} queue and enhancement of the software tools.

Traffic modeling

- A natural consequence of A.3 is the need for new traffic models that can accurately model both the \textsc{sr} and \textsc{lr} traffic properties. Furthermore, B.4 also implies that the emphasis should be on the \textsc{sr} traffic properties because of their determinative effect on the buffer performance. One promising approach is developing models based on fractal processes (see [Ryu96, RL96]).

Burst level analysis

- The main limitation of the \textit{Simputation} method described in Chapter 3 is the linear dependence of the execution time on the size of the state space of the underlying queuing model. The effect is particularly severe for prioritized queues. For instance, the state space of the \textsc{ldoll} queue increases polynomially with the buffer size, which virtually prohibits the \textsc{burst} level analysis of large-size \textsc{ldoll} queues. A promising approach is to use a heuristic approximation at the \textsc{cell} level that reduces the state space of the queuing model by neglecting the states with low equilibrium probabilities. The objective is to develop a heuristic that maximizes the execution time gains while introducing minimal errors.
6.2 Recommendations for future research

- As discussed in section 3.2.3, B.4 is only valid for medium-size buffers. The physical explanation of this result, as well as the infinite buffer approximation results, suggest that the performance impact of the self-similarity are likely to become more pronounced with the increase of the buffer size. Given the current trends in the industry, the switch buffers will continue to increase. Hence the analysis of the performance impact of self-similarity in large buffers is an area that needs more investigation in the future. One approach would be to use the heuristic version of the Simputation method discussed in the recommendation above.

- Another limitation of the Simputation method is the assumption of Bernoulli arrival process at the CELL level. The assumption is fairly reasonable in the case of a single ATM buffer, however for a tandem of buffers, the correlation between the subsequent cells on the same input link in the second stage can have a significant effect, especially for small fan-outs of the first stage buffer\(^1\). On the other hand the analysis of a tandem of LDOLL queues is important in order determine the performance impact of the LDOLL queuing after several stages. In particular one is interested in the LD cell loss probabilities. Since in this case the Simputation method is not applicable, except maybe for cases with large fan-out early stages, the most promising approach is to use importance sampling as an evaluation method.

Connection and path level analysis of the LDOLL queue

- Given the excellent performance of the LDOLL queue at the CELL and the BURST level, the next step is to investigate how to translate these performance improvements into an increased bandwidth utilization at the CONNECTION level. In other words, one needs an effective bandwidth approximation for the LDOLL queue. The main complication here is that the performance of e.g. the LD traffic is critically dependent on the load levels of the LL traffic. This further implies that the boundary of the associated admissible set is non-linear, i.e. the additivity property discussed in section 4.1 is not satisfied. The most promising approach is to work towards a definition of approximate linear bounds on the admissible set for the LDOLL queue in the spirit of [BW98a, BW98b]\(^2\).

- The last step in analyzing the LDOLL queue is to determine its effectiveness from a network point of view. In particular one is interested in its impact on the network topology and routing, but most importantly whether it can realize any substantial savings of costs at the PATH level. If we assume that the effective bandwidth approximation discussed above is already available, it should be reasonably easy to extend the VP dimensioning and VC routing algorithm of the design framework, and to implement changes in BANDIT. One could then use BANDIT to make a full comparison between an LDOLL enabled and an LDOLL disabled ATM network.

---
\(^{1}\)For an in-depth analysis of the so-called queuing network phenomena, see [vR97].

\(^{2}\)Very recent results that introduce a new notion of effective bandwidth for queues with priorities, suggest that under certain conditions, there exists a natural set of linear bounds on the admissible set, which allows application of stochastic-loss-network models for network design.
Tool enhancement

• Since this thesis includes only preliminary experiments with BANDIT, the logical next step is to perform a thorough quality assurance (QA) analysis of the tool and the associated design framework. Among other things, the analysis should provide some quantitative measure of the quality of the overall design process and pinpoint any potential shortcomings and weaknesses of the specific subproblem algorithms.

• Another straightforward extension is to implement the short- and medium-term operation modes that are discussed in section 4.4. This requires development of appropriate subproblem-solution algorithms at the local PATH sublevel (e.g. see algorithms described in [MMR96, VHW96]), and their implementation in BANDIT.

• One of the limitations of BANDIT is the long computation times of the design process. This is a consequence of the complexity of the design problem, which usually involves designing several inter-dependent subnetworks. One possible approach is to develop a distributed version of BANDIT that runs on several machines in parallel. A possible implementation, involving a master and slave architecture, is described in [Klo98].

• The main limitation of B-net is the excessive computation times, caused by the double-nested annealing algorithm, which forms the basis of the optimization procedure. A possible approach is to implement a faster inner-loop algorithm that would still produce satisfactory results.
Bibliography


153


BIBLIOGRAPHY


List of Symbols

This chapter contains a list of the most important mathematical symbols used throughout the thesis. The symbols are listed per section, in alphabetical order and are not repeated. If a symbol can not be found in the appropriate section, then it is either in one of the preceding sections or it is not listed. Greek symbols are sorted in alphabetical order according to their pronunciation.

General Notation

\( x \) \hspace{1cm} \text{a scalar}
\( X \) \hspace{1cm} \text{a random variable}
\( x \) \hspace{1cm} \text{a vector}
\( X \) \hspace{1cm} \text{a random vector variable}
\( X \) \hspace{1cm} \text{a matrix}
\( \Xi \) or \( \mathcal{R} \) \hspace{1cm} \text{a set}
\( \Pr\{\text{event}\} \) \hspace{1cm} \text{probability of occurrence of an event}
\( E[X] \) \hspace{1cm} \text{expectation value of a random variable}
\( \text{Var}[X] \) \hspace{1cm} \text{variance of a random variable}
\( X^T \) \hspace{1cm} \text{a transposition of a matrix } X
\( \min(x,y) \) \hspace{1cm} \text{minimum of } x \text{ and } y
\( \max(x,y) \) \hspace{1cm} \text{maximum of } x \text{ and } y
\( (\min(x,y))^+ \) \hspace{1cm} \text{max}(0,\min(x,y))
\( O(x) \) \hspace{1cm} \lim_{x \to \infty} f(x)/x = c < \infty
\( 1 \) \hspace{1cm} \text{a vector with all arguments unity}
\( \mathbb{N} \) \hspace{1cm} \text{the set of natural numbers}
\( \mathbb{N}_i \) \hspace{1cm} \text{the set of natural numbers greater than } i

Section 2.2

\( H \) \hspace{1cm} \text{Hurst parameter}
\( k \) \hspace{1cm} \text{the lag of } r(k)
\( \mu \) \hspace{1cm} \text{expectation value of a random process}
\( \sigma^2 \) \hspace{1cm} \text{variance of a random process}
\( r(k) \) \hspace{1cm} \text{autocorrelation function (see also the footnote on page 28)}
\( Y_t \) \hspace{1cm} \text{continuous-time stochastic self-similar process}
Section 2.3

\( A_n \) \( n \)-the customer interarrival interval
\( \eta^{-1} \) mean OFF period of an ON-OFF source
\( \lambda \) customer arrival rate at the CONNECTION level
\( \lambda(n) \) cell rate in the \( n \)-th point interval
\( \lambda(t) \) cell rate of a ON-OFF source at the BURST level
\( \Lambda(t) \) aggregate rate of an superposition of ON-OFF sources
\( N \) number of ON-OFF sources
\( N(t) \) number of arrivals in the interval \([0, t]\)
\( \omega_i(n) \) bit rate of a VBR source in the \( n \)-th frame
\( p_0(t) \) probability of no arrivals in the interval \([0, t]\)
\( \pi_i \) probability mass of the cell rate \( \lambda(i) \)
\( p_{\text{on}} \) probability that an ON-OFF source is active
\( \varrho(t) \) autocovariance function
\( R_{\text{peak}} \) ON cell rate of an BURST level ON-OFF process
\( \theta^{-1} \) mean ON period of an ON-OFF source
\( \tau_n \) a point process

Section 3.1

(\( \cdot \))_{\mathcal{R}} \text{ under retrieval policy } \mathcal{R}
\( A_{\text{LD/LL}}(t) \) number of LD/LL cells arrived at time slot \( t \)
\( c(\text{vecz}, s) \) cost function
\( \Delta \) threshold
\( \Phi \) arrival vector set
\( k \) inlet number
\( K(x) \) set of possible actions in state \( x \)
\( L_{\text{LD/LL}}(t) \) number of lost LD/LL cells at time \( t \)
\( \lambda_{\text{LD/LL}}(t) \) mean c cell arrival rate at time \( t \)
\( \text{LDi of LLi} \) LD or LL cells destined for outlet \( i \)
\( I_k(t) \) probability that the cell arrival on inlet \( k \) at time \( t \) is destined for outlet \( l \)
\( M \) number of queue outlets (servers)
\( N \) number of queue inlets
\( N_{\text{LD/LL}}(t) \) LD/LL queue occupancy at time \( t \)
\( \nu(\cdot) \) state evolution function
\( \Psi(\cdot) \) expected average cost per unit of time criterion
\( Q \) queue size in number of cells
\( \pi_x \) equilibrium state probabilities
\( p_x(t) \) probability of cell arrival on inlet \( k \) at time \( t \)
\( F_{xy}(t) \) one-step transition matrix
\( r_k(t) \) probability that the arrival on inlet \( k \) at time \( t \) is of LD type
\( \mathcal{R}(\Delta) \) LDOLL threshold policy
\( S_{\text{LD/LL}}(t) \) number of LD/LL cells served at time slot \( t \)
List of Symbols

\( W_{LD/LL}(t) \) \quad LD/LL sojourn time, i.e. LD/LL cell delay
\( \zeta(x) \) \quad arrival probability recursion

Section 3.2

\( B \) \quad buffer size
\( \omega_i(t) \) \quad cell-arrival rate on inlet \( i \) at time \( t \)
\( \pi(t) \) \quad state probability mass function at time \( t \)
\( T \) \quad scenario duration
\( T_j \) \quad BURST level load change interval

Section 4.2

\( a_l \) \quad average bandwidth carried on a VP link \( l \)
\( C_l \) \quad capacity of VP link \( l \)
\( C^\text{phy}_k \) \quad capacity of the physical link \( k \)
\( D_{\text{max}} \) \quad maximal number of VC and VP nodes on route \( r \) for service \( s \)
\( d_{\text{est}} \) \quad the effective bandwidth of traffic type \( s \) on link \( l \) in subnetwork \( v \)
\( \bar{d}_{\text{use}} \) \quad mean effective bandwidth on all routes of a \( (v, s, \sigma) \) traffic stream
\( \eta^{-1} \) \quad mean OFF period of the BURST level model of a \( (v, s, \sigma) \) traffic
\( f(r) \) \quad function returning the number of VC and VP nodes on route \( r \)
\( K \) \quad number of physical links in the network
\( L \) \quad total number of VP links in all subnetworks
\( L_{\text{use}}(\cdot) \) \quad blocking probability of of a service \( s \) on route \( r \) in subnetwork \( v \)
\( \lambda_{\text{use}} \) \quad mean call arrival rate of a \( (v, s, \sigma) \) traffic stream
\( \Lambda_{\text{use}} \) \quad mean arrival rate of calls from a \( (v, s, \sigma) \) traffic stream
\( m_{kl} \) \quad binary matrix element; equals 1 if VP link \( l \) traverses physical link \( k \)
\( N \) \quad number of nodes in the network
\( n^\text{phy}_l \) \quad Number of physical links traversed by a VP link \( l \)
\( p_{\text{rel}} \) \quad binary matrix element; equals 1 if there is a VP link between nodes \( i \) and \( j \) in subnetwork \( v \)
\( r \) \quad a VP route
\( \mathcal{R}(v, s, \sigma) \) \quad set of admissible routes for the \( (v, s, \sigma) \) traffic stream
\( \rho_{\text{peak}} \) \quad peak rate of the BURST level model of a \( (v, s, \sigma) \) traffic
\( \rho_{\text{use}} \) \quad traffic intensity of a \( (v, s, \sigma) \) stream on route \( r \)
\( \bar{\rho}_{\text{use}} \) \quad total traffic intensity of a \( (v, s, \sigma) \) stream on all routes \( r \)
\( p_{\text{use}} \) \quad Bernoulli probability of the admission control function
\( s \) \quad number of traffic types (services) in the network
\( \sigma \) \quad an origin-destination pair of nodes
\( \theta^{-1} \) \quad mean ON period of the BURST level model of a \( (v, s, \sigma) \) traffic
\( V \) \quad number of VP subnetworks configured on top
\( (v, s, \sigma) \) \quad traffic stream of type \( s \), from origin-destination pair \( \sigma \),
carried by subnetwork \( v \)

\( \mu_{vsr}^{-1} \)
mean duration of calls from the \((v, s, r)\) traffic stream

\( Y_{sr} \)
average number of VPs involved in the call-setup phase

\( Z_{B}(\cdot) \)
call blocking cost function

\( Z_{C}(\cdot) \)
switching and setup cost function

\( \zeta_{S} \)
setup unit cost per call per VP involved in the call-setup phase

\( \zeta_{VC} \)
VC switching cost per unit of bandwidth

\( \zeta_{VP} \)
VP switching cost per unit of bandwidth

**Section 4.3**

\( B_{sl} \)
blocking probability of service \( s \) calls on link \( l \)

\( c_{sl} \)
IMPLIED cost of service \( s \) calls on link \( l \)

\( e_{usr} \)
revenue earned per call per unit of time, by calls

of type \( s \) on route \( r \)

\( \nu_{sl} \)
reduced load of service \( s \) calls on link \( l \)

\( \nu_{sl,r} \)
reduced load of service \( s \) calls on route \( r \) offered to link \( l \)

**Section 4.5**

\( \epsilon \)
the most stringent QoS requirement (lowest loss probability)
of all services clustered in the same VP subnetwork

\( \epsilon_{s} \)
the loss probability QoS requirement of service \( s \)

\( \text{hops}(r) \)
the number of hops in the route \( r \)

\( \text{min-hop}(\sigma) \)
the number of hops in the shortest route for the

source-destination pair \( \sigma \)

\( \text{NrHops} \)
the maximal length of a route \( r \) in VP/VC hops for service \( s \)

\( \text{Plus} \)
the permissible number of additional hops to the shortest route,

to provide for additional slack in the routing set

**Section 5.2**

\( d_{i}^{q} \)
cost of an CLS with capacity \( q \)

\( e_{ij}^{c} \)
cost of connecting nodes \( i \) and \( j \) with a PVC of capacity \( c \)

\( f_{ij}^{cd} \)
mean traffic flow from source \( o \) to destination \( d \),
on the link between nodes \( i \) and \( j \)

\( f_{ij} \)
total traffic traversing link \( ij \)

\( \lambda_{ij} \)
mean traffic load from node \( i \) to node \( j \)

\( N \)
set of network nodes

\( Q \)
set of CLS capacities

\( x_{ij}^{c} \)
binary variable; equals 1 if link of capacity \( c \) exists
between nodes \( i \) and \( j \)

\( y_{i}^{q} \)
binary variable; equals 1 if a CLS with capacity \( q \)
is present at node \( i \)

\( T_{\text{max}} \)
maximum tolerable network-wide delay
List of Abbreviations

Because of the importance of the traffic classification in this thesis the abbreviations of the four traffic classes are listed first. The rest of the abbreviations are listed in alphabetical order.

LX  Low Delay and Low Loss
LD  Low Delay
LL  Low Loss
HX  High Loss and High Delay

AAL  ATM Adaptation Layer
ABR  Available Bit Rate
ARIMA  Auto-Regressive Integrated Moving Average
ATM  Asynchronous Transfer Mode
BANDIT  Broadband ATM Network Design Tool
BISDN  Broadband Integrated Services Digital Network
BOP  Buffer Overflow Probability
CAC  Call Acceptance Control
CB  Common Buffer
CBR  Constant Bit Rate
CDF  Cumulative Distribution Function
CE  Cell Enqueuer
CFA  Capacity and Flow Assignment problem
CLP  Cell Loss Priority bit
CLS  Connectionless Server
CP  Complete Partitioning
CPU  Computer Processing Unit
CR  Cell Receiver
CS  Cell Server
CS  Complete Sharing
CT  Cell Transmitter
D-BMAP  Discrete time Batch Markovian Arrival Process
DQDB  Distributed Queue Dual Bus
DVE  Distributed Virtual Environment
fARIMA  Fractional ARIMA
FD  Flow Deviation
FDDI  Fiber Distributed Data Interface
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FEC</td>
<td>Forward Error Correction</td>
</tr>
<tr>
<td>FIFO</td>
<td>First In First Out</td>
</tr>
<tr>
<td>FSM</td>
<td>Finite State Machine</td>
</tr>
<tr>
<td>FTP</td>
<td>File Transfer Protocol</td>
</tr>
<tr>
<td>GFC</td>
<td>Generic Flow Control</td>
</tr>
<tr>
<td>GII</td>
<td>Global Information Infrastructure</td>
</tr>
<tr>
<td>GMCBP</td>
<td>Generally Modulated Compound Bernoulli Process</td>
</tr>
<tr>
<td>GoS</td>
<td>Grade of Service</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HEC</td>
<td>Header Error Control</td>
</tr>
<tr>
<td>HX</td>
<td>High Loss and Delay class</td>
</tr>
<tr>
<td>IB</td>
<td>Input Buffer</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>III</td>
<td>International Information Infrastructure</td>
</tr>
<tr>
<td>IOS</td>
<td>Internet Operating System</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>ISDN</td>
<td>Integrated Services Digital Network</td>
</tr>
<tr>
<td>ISH</td>
<td>Information Superhighway</td>
</tr>
<tr>
<td>ITU-T</td>
<td>International Telecommunication Union, Telecommunication Standardization Sector</td>
</tr>
<tr>
<td>IWU</td>
<td>Inetworking Unit</td>
</tr>
<tr>
<td>JPEG</td>
<td>Joint Photographers Experts Group</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>LD</td>
<td>Low Delay class</td>
</tr>
<tr>
<td>LDOLL</td>
<td>Low Delay Or Low Loss</td>
</tr>
<tr>
<td>LL</td>
<td>Low Loss class</td>
</tr>
<tr>
<td>LP</td>
<td>Linear Programming</td>
</tr>
<tr>
<td>LRD</td>
<td>Long-Range Dependence</td>
</tr>
<tr>
<td>LX</td>
<td>Low Delay and Low Loss class</td>
</tr>
<tr>
<td>MAC</td>
<td>Medium Access Control</td>
</tr>
<tr>
<td>MAN</td>
<td>Metropolitan Area Network</td>
</tr>
<tr>
<td>MID</td>
<td>Multiplexing Identifier</td>
</tr>
<tr>
<td>MOD</td>
<td>Music on Demand</td>
</tr>
<tr>
<td>NNI</td>
<td>Network-to-Network Interface</td>
</tr>
<tr>
<td>OAM</td>
<td>Operation and Maintenance</td>
</tr>
<tr>
<td>OB</td>
<td>Output Buffer</td>
</tr>
<tr>
<td>OC3</td>
<td>Optical Carrier (level) 3 (SONET rate of 155.52 Mbits/s)</td>
</tr>
<tr>
<td>OOP</td>
<td>Object Oriented Programming</td>
</tr>
<tr>
<td>OSI</td>
<td>Open System Interface</td>
</tr>
<tr>
<td>PDH</td>
<td>Plesioschronous Digital Hierarchy</td>
</tr>
<tr>
<td>PDU</td>
<td>Protocol Data Unit</td>
</tr>
<tr>
<td>PHY</td>
<td>Physical layer</td>
</tr>
<tr>
<td>PM</td>
<td>Physical Medium</td>
</tr>
<tr>
<td>POTS</td>
<td>Plain Old Telephone Service</td>
</tr>
<tr>
<td>PS</td>
<td>Partial Sharing</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Definition</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>PTI</td>
<td>Payload Type Identifier</td>
</tr>
<tr>
<td>PVC</td>
<td>Permanent Virtual Connection</td>
</tr>
<tr>
<td>QA</td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RSVP</td>
<td>Resource Reservation Protocol</td>
</tr>
<tr>
<td>SA</td>
<td>Simulated Annealing</td>
</tr>
<tr>
<td>SDH</td>
<td>Synchronous Digital Hierarchy</td>
</tr>
<tr>
<td>SONET</td>
<td>Synchronous Optical Network</td>
</tr>
<tr>
<td>SRD</td>
<td>Short-Range Dependence</td>
</tr>
<tr>
<td>SVC</td>
<td>Switched Virtual Connection</td>
</tr>
<tr>
<td>TC</td>
<td>Transmission Convergence</td>
</tr>
<tr>
<td>TU-Delft</td>
<td>Delft University of Technology</td>
</tr>
<tr>
<td>UAA</td>
<td>Uniform Asymptotic Approach</td>
</tr>
<tr>
<td>UBR</td>
<td>Unspecified Bit Rate</td>
</tr>
<tr>
<td>UNI</td>
<td>User-to-Network Interface</td>
</tr>
<tr>
<td>UPC</td>
<td>Usage Parameter Control</td>
</tr>
<tr>
<td>VBR</td>
<td>Variable Bit Rate</td>
</tr>
<tr>
<td>VC</td>
<td>Virtual Channel (Connection)</td>
</tr>
<tr>
<td>VCI</td>
<td>Virtual Channel Identifier</td>
</tr>
<tr>
<td>VLSI</td>
<td>Very Large Scale of Integration</td>
</tr>
<tr>
<td>VOD</td>
<td>Video on Demand</td>
</tr>
<tr>
<td>VP</td>
<td>Virtual Path</td>
</tr>
<tr>
<td>VPI</td>
<td>Virtual Path Identifier</td>
</tr>
<tr>
<td>VR</td>
<td>Virtual Reality</td>
</tr>
<tr>
<td>WAN</td>
<td>Wide Area Network</td>
</tr>
<tr>
<td>WLL</td>
<td>Wireless Local Loop</td>
</tr>
<tr>
<td>WWW</td>
<td>World Wide Web</td>
</tr>
</tbody>
</table>
Samenvatting (Summary in Dutch)

De belangstelling voor de Global Information Infrastructure (GII) heeft in de laatste paar jaar drastisch gestegen voornamelijk door de enorme populariteit van de Internet en de nieuwe ontwikkelingen van netwerk technologieën zoals Java. Dit proefschrift beschrijft efficiënte methoden voor ontwerpen van ATM netwerken die de basis van de toekomstige GII vormen.

Van de ATM netwerken wordt verwacht dat ze op een efficiënte manier spraak, video, data en multimedia diensten op een gemeenschappelijke infrastructuur kunnen verenigen. Enerzijds is de efficiënte gebruik van de netwerk middelen van groot belang omdat de netwerk carriers miljarden dollars per jaar in het vergroten van de netwerk capaciteit moeten investeren, om de enorm snelle gooi van de netwerk verkeer aan te kunnen. Anderzijds de ondersteuning van een grote aantal diensten met verschillende QoS eisen impliceert een lage utilisatie van de netwerk middelen. Dus het ontwerp van ATM netwerken wordt een werkelijke uitdaging.

De aanpak van de proefschrift is gebaseerd op stochastische modellering. De ATM netwerk is gemodelleerd als een netwerk van wachtrijen, waar de aankomende verkeer wordt geregereerseerd met geschikte verkeersmodellen en de netwerk nodes als wachtrijen die onderling met servers van vaste snelheid gekoppeld zijn. Om een efficiëntere gebruik te maken van de netwerk middelen wordt de netwerk verkeer in vier verschillende klassen verdeeld: Low Loss and Delay klas, Low Delay klas, Low Loss klas and High Loss and Delay klas (LX, LD, LL and HX respectievelijk). De analyse heeft aangetoond dat de grootste winst wordt behaald als de LX and HX klassen apart worden behandeld en de LD en LL klassen van een gemeenschappelijk netwerk gebruik maken.

De analyse volgt een multi-laag aanpak. De single-link-analyse beschouwt de prestatie van een alleenstaande ATM link, terwijl de netwerk-analyse beschouwt de hele ATM netwerk. De single-link-analyse richt zich op de zogenaamde LDOLL wachtrij, een nieuwe wachtrijstrategie die de prestatie van beide LD en LL klassen verbetert. De prestatie van beide enkel- en multi-uitgang LDOLL wachtrij wordt uitgebreid geanalyseerd onder verschillende omstandigheden. De prestatie van de nieuwe strategie is aanzienlijk beter dan de standaard FIFO (First-In-First-Out) strategie voor vele verschillende buffer groottes, verkeersmodellen en tijdschalen.

De single-link-analyse beschouwt ook de invloed van de recent ontdekte self-similarity eigenschap van de netwerk verkeer op de buffer prestatie en eveneens de betrouwbaarheid van de traditionele verkeersanalyse methoden in het licht van de nieuwe ontdekking. De
resultaten duiden aan dat de self-similarity eigenschap allen gering impact heeft op de buffer prestatie en dat de traditionele methoden nog steeds van toepassing zijn.

De netwerk analyse richt zich op het ontwerpen van ATM netwerken en in het bijzonder, op de effecten van de ATM multi-layer structuur op de netwerk efficiëntie. De doel is om een hogere utilisatie te bereiken door de virtuele sub-netten optimaal te configureren. Als resultaat wordt er een nieuwe ontwerp algoritme en een software tool ontworpen, die voor gegeven verkeer belasting en QoS eisen een sub-net configuratie met maximale utilisatie levert.

De proefschrift eindigt met een samenvatting van de gevonden resultaten, originele contributies en aanbevelingen voor verder onderzoek.
Acknowledgements

In the course of the four years that I spent working on this thesis a great number of people have helped me in various ways. I want to thank them all for their help and support. Since it is impossible to name them all in such a short space, I will mention just a few that were invaluable to my work.

I owe very much to my mentor Frits Schoute for his inspiring personal guidance through this four-year odyssey. I thank him for all his support and continued interest in my work, even at times when he was fully engaged in different projects. I found our collaboration most inspiring, because of his endless enthusiasm, his genuine creativity and his ability to see things differently from others. I am also thankful for his patience and his confidence in me, when things did happen later than originally planned.

I thank all my colleagues at the Telecommunications and Traffic Control Systems Group, for their companionship and an international atmosphere that made me feel at home. Special thanks are due to my closets mates Dušan Matić, Marco Moretti and Frank van der Wijk, for being good friends, for making me laugh and for making life bearable when times were tough.

I am grateful to all student members of the A-TeaM that I have worked with, for their interest in my research, for the many contributions they have made to it and for all the fun that we had together. I am specially grateful to Richard van den Berg and Thijs Harleman, whose research results are used in chapters 2 and 3, and particularly to Gerard Klop, who spent a considerable amount of time after his graduation on fixing bugs in BANDIT and conducting experiments, which significantly added to the quality of chapter 4.

I thank Mirjam Nieman for correcting the English text, Frits Schoute for proofreading several different versions of the manuscript and Darko Stavrov and Irena Talaganova for designing the cover.

Last but not least I thank my family for all their love and support during the course of my studies, and especially my wife Suzana, who has contributed to this thesis almost as much as I did, by taking over a lion share of the load from my shoulders and by being there when I needed most.

173
Curriculum Vitae

Borut Stavrov was born in Skopje, Macedonia, on March 22, 1969. He had his secondary education at the “Orce Nikolov” School in Skopje. From 1986 to 1991 he studied Electrical Engineering at the “St. Kiril and Metodij” University in Skopje, specializing in telecommunications engineering.

In 1992 he enrolled the Faculty of Electrical Engineering at the Delft University of Technology, the Netherlands, where he worked towards his M.Sc. degree. In March 1994 he received his M.Sc. in Electrical Engineering from the Delft University of Technology. His research focused on the analysis and design of novel telecommunication switching techniques.

From April 1994 until May 1998 he worked as a research assistant (Assistent in Opleiding) at the Telecommunications and Traffic Control Systems Group of the former Faculty of Electrical Engineering at the Delft University of Technology. The results described in this thesis follow from the research efforts carried out in this four year period.

In June 1998 he joined TIBCO Finance Technology Inc., where he currently works on analysis and design of networking software (middleware) technologies and applications.