HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Electrical and Communication Engineering
Networking Laboratory

Esa Hyytiä

Dynamic Control of All-Optical WDM Networks

Licentiate thesis

Supervisor: Professor Jorma Virtamo, HUT

Preface

This licentiate thesis has been written in the Networking Laboratory of Helsinki University of Technology. The research was done within the COST 257 project funded by Nokia, Sonera and Tekes.

First I want to thank my supervisor professor Jorma Virtamo for his invaluable guidance during the last four years. I am also thankful for the second reviewer of the thesis, Ph.D. Samuli Aalto, for numerous corrections and advice when finishing the thesis. Also the whole personnel of the Networking Laboratory earns special thanks for providing a good atmosphere and good facilities for the research.

This thesis has also been financially supported by a grant from the Nokia Foundation.

Espoo, February 12, 2001

Esa Hyytiä

Abstract

In this work the dynamic routing and wavelength assignment problem (D-RWA) in all-optical WDM network is studied. The problem is approached in the framework of the Markov Decision Process theory. In practice the optimal policy cannot be exactly calculated due to the huge size of the state space, but heuristic algorithms can come quite close to the optimal policy. The main contribution of the work is the application of so called first policy iteration to D-RWA.

In the first policy iteration one tries to improve a given policy. This is done by considering the expected future costs that consist of the immediate cost of the chosen action and the relative cost of the next state. Immediate costs are known while the relative costs are not. Furthermore, the relative costs of the states cannot be obtained for every state due to the astronomical state space size. However, at each decision epoch the relative values are only needed for a small set of possible actions. Thus, instead of trying to solve them exactly their values are estimated by simulations.

The work consists of four parts. In first part a brief introduction to all-optical WDM networks is given. The second part is a survey of the static routing and wavelength assignment problem, where the problem is described together with some heuristic RWA algorithms. The third part of the thesis considers the dynamic routing and wavelength assignment problem and contains, among other things, a description how the first policy iteration can be applied to D-RWA problem, together with some simulation results. The fourth part contains a brief survey of the important restoration and protection aspects in all-optical networks.

Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

ANSI	American National Standards Institution, 10
AON	All-Optical Network, 5
ATM	Asynchronous Transfer Mode, 10
AWG	Arrayed Waveguide Gratings, 6
CDC	Cycle Double Cover (conjecture), 100
DCA	Distinct Channel Assignment, 17
DFB	Distributed-FeedBack (laser), 6
DWDM	Dense Wavelength Division Multiplexing, system where several (typically 40 or 80) optical channels near each other are transmitted in the same fibre, 1, 5
EDFA	Erbium Doped Fibre Amplifier, 8
ETSI	European Telecommunications Standards Institute, 10
IETF	Internet Engineering Task Force, 9
IP	Internet Protocol, 11
ITU	International Telecommunication Union, 5
Laser	Light Amplification by Stimulated Emission of Radiation, 6
LDC	Linear Divider Combiner, an optical network node element, 12
LLN	Linear Lightwave Network, 12
LRN	Logically Routed Network, 12
LSN	Logical Switching Node, 12
MDP	Markov Decision Process, 33, 101
MPLS	Multi Protocol Label Switching, 11
NMS	Network Management System, 83
NP	Nondeterministic Polynomial-time, 93
NP-complete	Subset of class NP problems which are considered very hard to solve, 93
OCDC	Orientable Cycle Double Cover (conjecture), 100
ONN	Optical Network Node, 13
OXC	Optical Cross-Connect, 81
RWA	Routing and Wavelength Assignment, 17
SDH	Synchronous Digital Hierarchy, European counterpart of SONET, 10
SONET	Synchronous Optical Network, American counterpart of SDH, 10
STM	Synchronous Transport Module, 10
TCP	Transmission Control Protocol, 9
UDP	Universal Datagram Protocol, 9
WDM	Wavelength Division Multiplexing. Technology where several optical signals using different wavelengths share the same media, 5
WIXC	Wavelength Interchange Cross-Connect, a cross-connect which is capable to do a wavelength translations, 7, 22
WRN	Wavelength Routed Network, a network where routing in the nodes is done per wavelength channel basis, 11
WSXC	Wavelength Selective Cross-Connect, a cross-connect which is not capable to do a wavelength translations, 7, 22

Notation

$\mathcal{K}$	the set of traffic classes
λ_k	arrival rate of class-k connection requests (Poisson process)
μ_k	the connection holding time parameter, exponentially distributed
w_k	revenue rate of class-k connection
β_k	the average loss of missed class-k connection, i.e. $\beta_k = w_k/\mu_k$.
$\mathcal{S}$	the state space of the system.
α	policy, $\alpha : \mathcal{S} \times \mathcal{K} \to \mathcal{S}$.
$\mathcal{A}_{i,k}$	possible states the system can move to when class-k arrival occurs in state i, including the current state i
r_i	revenue rate in state i, explicitly written $r_i(\alpha)$ means the revenue rate in state i under policy α.
$r(\alpha)$	average revenue rate of the system under policy α.
$c_t(i)$	expected cost rate at time t when system initially starts from state i, $c_t(i) = \mathrm{E}[r_{X_t} \| X_0 = i]$.
v_i	relative cost of state i, the difference in costs when system starts from state i instead of equilibrium: $v_i = \int_0^\infty (c_i(t) - c)\ dt$, where c is the average cost rate of the system in equilibrium and $c_i(t)$ is the expected cost rate of the system at time t when it is in state i at time $t = 0$. Furthermore, $v_i(\alpha)$ means relative cost of state i under policy α.
$\mathbf{Q}$	transition intensity matrix of Markov process
W	number of wavelengths available
κ	adjustable parameter for accepting an alternative policy in the first policy iteration
N	the number of simulation replications in the first policy iteration approach

Chapter 1

Introduction

The rapid growth of the Internet traffic has been the driving force for faster and more reliable data communication networks. Wavelength division multiplexing (WDM) is a very promising technology to meet these ever increasing demands. In a WDM network several optical signals are sent on the same fibre using different wavelength channels. Sometimes the term dense wavelength division multiplexing (DWDM) is used to distinguish the technology from the broadband WDM systems where two widely separated signals (typically 1310 nm and 1550 nm) share a common fibre. In DWDM up to 40 or 80 signals are combined in the same fibre.

Traditionally only a small fraction of the fibre capacity is in use, but by using WDM it is possible to exploit this huge capacity more efficiently. The possibility to use the existing fibres more efficiently makes WDM a very attractive alternative commercially, as it is often very expensive to install new fibres in the ground. This is the case especially in densely populated areas like cities, where fibres must be dug under streets etc.

WDM technology has been recognized as one of the key components of the future networks. The commercialization of WDM technology is progressing rapidly. Especially important for the development of the WDM technology was the invention of the optical fibre amplifier in 1987 (Erbium doped fibre amplifier, EDFA). The optical fibre amplifier is a component capable to amplify several optical signals at the same time without converting them first to electrical domain (opto-electronic amplification). It is also worth noting that EDFA can be used to amplify signals of different bit rates and modulations. Other important WDM components include lasers, receivers, wavelength division multiplexers, wavelength converters, optical splitters and tunable filters among others.

There is also a wide interest towards the optical networking in academic community as it offers a rich research field for scientists from the component level up to the network protocols. Also popular scientific magazines have published articles about the optical networks [Sti01, BGD01, Blu01].

The first WDM systems being currently deployed are point-to-point systems, where the gain is simply a larger link capacity. Many companies are already offering such solutions in their product line. The next step in backbone networks will probably be the wavelength routed networks (WRN). WRN is a very scalable[1] network and can exploit the vast bandwidth of the optical fibre throughout the network more efficiently. WRN also allows quick restoration schemes and modifications of underlying network without reconfiguration of upper layers. Hence the optical layer is used to build the so called virtual topology over the physical network for the logical layer (e.g. ATM). The virtual topology can remain the same even if the physical networks changes for some reason like failure in some part of the network.

This work focuses on the dynamic control of fully optical wavelength routed WDM network, in other words how the arriving connections, i.e. lightpath requests, should be configured into the network in order to maximize the expected revenues. The networks considered in this thesis are arbitrary mesh networks instead of regular structures like ring networks. The optimization goal is to minimize some infinite time horizon cost function such as the blocking probability. The analysis is done within the framework of Markov Decision Processes (MDP). The MDP theory is widely applied in the field of telecommunications, e.g. in the traditional circuit switched networks.

1.1 Outline of the work

In Chapter 2, the WDM technology is briefly discussed to give necessary information about the background of the problem. The key components of a WDM network are presented. Also the current technologies like SONET, SDH and ATM are introduced. Then, in the next chapter, configuration of static traffic is formulated in a few alternative ways. Generally the problem is to configure a given set of lightpath requests into the network. The chapter also contains some bounds for the number of wavelengths required to satisfy given lightpath requests. Mathematically the problem is a resource allocation problem.

Chapter 4 first defines the dynamic routing and wavelength assignment (D-RWA) problem. Then an exact algorithm to solve D-RWA problem within the MDP theory framework is presented. Due to the prohibitive size of the state space this is not practical and some suboptimal solution must be used. This means using heuristic policies. The main content of the chapter is the first iteration policy, which is used to improve the current (heuristic) policy. The treatment is also extended outside Markovian traffic models.

In Chapter 5 the presented heuristic algorithms are evaluated. The first policy iteration approach is tested and reported to perform well. Then some con-

[1] In comparison, broadcast and select optical networks do not scale well with the number of end systems.

clusions about the heuristic policies and the problem are drawn. Finally, in Chapter 6 the aspects of restoration and survivability of WDM-network are briefly discussed. The appendices contain background material mainly from the graph theory and the MDP theory. Also one appendix contains a brief description of the simulator program used to obtain the simulation results presented in Chapter 5.

Chapter 2

Introduction to WDM

Wavelength division multiplexing (WDM) is a promising technology for future all-optical networks. In WDM several optical signals using different wavelengths share the same fibre. The capacity of such fibre links can be huge, even terabits per second. So essentially the optical spectrum is used more efficiently. Routing in the network nodes is based on wavelengths of incoming signals [Muk97, RS98, Wil97]. Currently the WDM technology is used to increase the capacity of optical links where at the end of each link the signal is converted back to electrical domain. But the technology is progressing towards transparent all-optical networks where the signal is routed through the network in the optical domain.

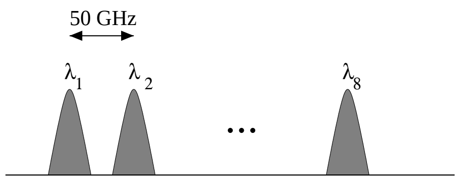

Figure 2.1: The optical spectrum and 8 wavelength channels.

The International Telecommunication Union (ITU) has standardized the use of the wavelength channels in a WDM link in standard G.692 (see [ITU98]). The channel spacing is proposed to be 50 GHz or 100 GHz around the reference frequency of 193.10 THz, as depicted in Fig. 2.1. 193.10 THz corresponds to about 1550 nm, hence the proposal is meant for the 1540 nm - 1560 nm pass band of the optical fibre.

2.1 Components of WDM-Network

During recent years lots of effort has been put into the development of better optical components to enable all-optical WDM-networks (AON) [EM00b]. The most important components are light sources, tunable optical filters, optical switches and of course the fibre. Different components are briefly presented in the following sections.

2.1.1 Light Sources

One important element of an optical system is the light source. For communication purposes a good light source should be quickly tunable with a wide range of wavelengths. To make a component also commercially attractive low power consumption and low price are vital parameters [EM00b]. The time scale of tuning depends on case, with the optical packet switching the requirements are somewhere between microseconds and nanoseconds while with circuit switched WDM-networks the time scale is slower.

Here is a list of several candidates [EM00b, Muk97]:

1. Mechanically tuned lasers
2. Acousto-optically and electro-optically tuned lasers
3. Injection current tuned lasers
4. Switched sources
5. Array sources (using arrayed waveguide gratings (AWG) or distributed-feedback (DFB) lasers)

Mechanically tuned lasers, for example, have a tuning time of the order of milliseconds and are thus too slow for packet switched optical networks. Generally the choice between different light source types depends on the application and the two most important parameters for light sources are the tuning time and the tuning range.

2.1.2 Tunable Filters

A tunable optical filter is also an important part of the optical network. Many promising approaches have been studied including Fabry-Perot, acousto-optic, electro-optic and liquid crystal Fabry-Perot filters (see e.g. [EM00b, Muk97]) The filters have two important parameters dealing with the performance: tuning range and tuning time. The tuning ranges are from around 10 nm up to 500 nm, while the tuning time is from nanoseconds up to 10 milliseconds.

2.1.3 Optical Switches

The optical switch, or optical cross-connect (OXC), is a device which can be dynamically configured to connect given input ports to any of the output ports. The optical switches can be classified according to how flexible they are [SB99]:

1. *A non-blocking switch* means any connection pattern can be realized by reconnection of some or all of the current connections.

2. *Wide-sense non-blocking switch* is a switch which can, with careful configuration, add any new connection without interrupting previously configured connections through the switch.

3. *Strict-sense non-blocking switch*, on the other hand, means that a simple configuration strategy allows adding new connections to the switch any time without interrupting any of the current connections.

Clearly the number of elements and device complexity grows at the same time as the flexibility. This means a trade-off between hardware complexity and management complexity.

Wavelength channels

In WDM-networks each fibre contains W wavelength channels, and thus the optical switches should be capable to treat channels individually. The optical cross-connects used in WDM-networks can be divided into two categories. A wavelength selective cross-connect (WSXC) is a device capable to configure any given input λ-channel from arbitrary input port to a given output port (using the same wavelength).

Wavelength translation (conversion) is an operation where an incoming signal using λ_1-channel is converted to another channel λ_2 at the output port. Wavelength interchange cross-connect (WIXC), depicted in Fig. 2.2, is a more advanced device than WSXC which can manipulate wavelengths of the signals as well, i.e. an incoming signal can emerge from the switch using another wavelength. Hence, such a device can configure any λ_1-channel from any input port to any output port using λ_2-channel, i.e. it is capable of doing wavelength translations as well. Clearly a WIXC device is more complex than WSXC, but it also gives more flexibility in the configuration of the network, and hence leads to more efficient use of the network resources.

Note that both WSXC and WIXC are devices where every input channel is connected to no more than one output-channel (permutation switch).

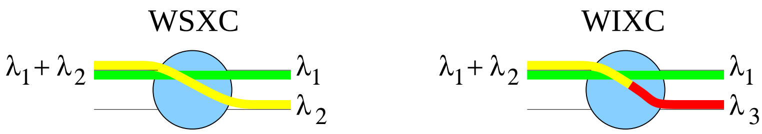

Figure 2.2: The basic components of the wavelength routed network. Wavelength selective cross-connect (WSXC) routes incoming signals per wavelength basis, while wavelength interchange cross-connect (WIXC) has also capability to perform wavelength translations.

2.1.4 Wavelength Conversion

Wavelength conversion, as noted in the previous section, allows more efficient use of the network resources. The reason is that without it so called wavelength-continuity constraint has to be satisfied, i.e. the lightpath reserves the same wavelength all the way along the route. Hence, even if there are free channels available in every link of the network, some connections may not be configured unless wavelength conversion is possible in some of the nodes.

Again, an easy solution is to do the opto-electronic wavelength conversion where the optical signal is first converted to the electric domain and then reproduced in the optical domain at a different wavelength. The drawback with this approach is the limited bit rate of electronics.

An other approach is to do the conversion in the optical domain. Suggested solutions include using the four-wave mixing and fibre nonlinearities, and cross modulation with active semiconductor devices. An up-to-date survey on wavelength conversion can be found in [EM00a].

2.1.5 Optical Amplifiers

The attenuation of optical signals is low in comparison with electrical signals. Still long-distance links may need amplifiers in order to operate properly. The traditional way to solve the problem is to convert the signal back to electrical domain for amplification and retransmit it optically. This approach, however, requires knowledge of the used bit rate and modulation. A new solution is to use amplifiers operating totally in the optical domain. In particular, the erbium doped fibre amplifier (EDFA) operating at 1540 nm region has proven to be an excellent choice for the WDM systems. The amplifier is transparent to used coding and bit-rate, and thus suits well to all-optical framework. Also a similar amplifier for the 1300 nm region has been built using praseodymium instead of erbium.

2.2 Today's Technology and Solutions

In the field of data communications things change rapidly. What is the state of the art technology today is probably old technology tomorrow. In this chapter some idea about the limitations of the current optical technology is given. It will serve as a background for the stochastic models and optimization techniques presented in the later chapters.

2.2.1 Internet Protocol, IP

The Internet Protocol (IP) has been a massive success story during the last ten years. The dawn of the Internet was back in the 70's when Arpanet was created. Arpanet was funded by DARPA, the agency of U.S. Department of Defense (DoD) in charge of advanced research projects [Hui95]. Since then the Internet has grown to a worldwide network and the number of computers connected to it grows extremely fast. Nowadays the Internet standards are defined within IETF (the Internet Engineering Task Force), which coordinates the standardization work in the Internet community. The requests for comments, shortly RFCs, form the backbone of documents describing how things are done in the Internet.

The IP defines a packet based way to communicate over a heterogeneous environment including slow modem links as well as high capacity backbone routes. Generally the IP network is a best effort network, with no guarantees for any quality of service. In case there is too much traffic, packets will be lost. The related layer 4 protocols the applications use are UDP (Universal Datagram Protocol) and TCP (transmission control protocol), and are thus built upon the IP. The UDP is a connectionless protocol to send an IP packet. Nothing is guaranteed, the packet may be lost without any feedback. It is left to the application to solve possible problems. The TCP, on the other hand, is a connection oriented protocol where the stream of bytes is guaranteed to reach the destination in correct order (or the failure is reported to the both ends). TCP/IP also includes mechanisms for the congestion control with slow start where the transmission rate is slowly increased until a packet is lost and the transmission speed is cut to a half. Hence, TCP/IP is scalable to different transmission speeds.

Many of the current popular applications including E-mail, WWW, ftp and telnet rely on TCP/IP, while UDP is used in some applications like NFS (network file system) and real-time applications.

2.2.2 SDH and SONET

SDH stands for the synchronous digital hierarchy and is a widely used transmission system in Europe. SONET, synchronous optical network, is its American counterpart. SDH is defined by the European Telecommunications Standards Institute (ETSI), while the SONET is defined by the American National Standards Institution (ANSI). These standards define the line rates, coding schemes, bit-rate hierarchies, restoration and network management. Equipment from different vendors can be used together and network operators get more freedom in building their networks.

Both systems use a small time frame containing a header and a payload as a basic building block. Higher transmission rates are obtained by byte-interleaving the basic time frames[1]. Almost all of the processing is done digitally. Optical signals in SONET are denoted with OC-x, where x defines the bit-rate. For example, OC-48 signal means 2.5 Gbit/s and OC-192 signal corresponds to about 10 Gbit/s transmission speed. The SDH counterparts for the OC-signals are STM-signals where STM-1 designates transport rate of 155.52 Mbps and other STM-4, STM-16 and STM-64 have similarly 4, 16 and 64 times higher transmission speed than the STM-1 signal. Thus, STM-64 and OC-192 both have a transmission rate of 10 Gbps.

2.2.3 ATM

The asynchronous transfer mode (ATM) has received a lot of research interest during the last years as well. It is a cell-oriented[2] switching and multiplexing technology. In the ATM concept the basic building block is a 53-byte long cell divided into 5 bytes long header and 48 bytes long payload (see Fig. 2.3). Originally ATM was developed to support different kinds of traffic (service classes) with different quality of service (QoS) requirements and intended to be used up to the end nodes, but currently it is mainly used in the backbone networks.

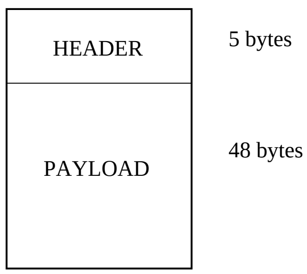

Figure 2.3: ATM-cell, header contains information about VP and VC and payload the actual data.

[1]Hence the name synchronous as whole network is assumed to work synchronously
[2]Cell is a fixed length packet

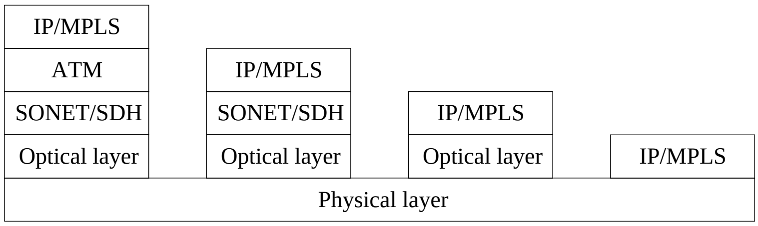

Figure 2.4: Possible alternatives for IP-over-WDM solutions.

ATM is a *connection oriented* network technology, i.e. the connections must be set up before information can be transferred, and later released. Two important concepts with ATM technology are virtual paths (VP) and virtual channels (VC). The virtual paths are used to form a virtual topology over the physical ATM network. Each virtual path carries one or more virtual channels which are statistically multiplexed in the virtual path.

2.3 Evolution of WDM Technology

Telecommunication field is full of standards defining different layers for the whole infrastructure. In the past the end users were people making phone calls or using fax machines etc. But now it has become very clear that in the future almost all the traffic will be IP-based. The evolution will go towards IP-over-WDM networks, where several alternative approaches have been proposed [BRM00, GDW00]. In Fig. 2.4 some of the possible layering alternatives are depicted. Each additional layer brings naturally some extra overhead to the transmission. Hence, the standard IP over ATM over SONET / SDH over WDM mapping can be considered as an inefficient solution. The other extreme is a direct IP / MPLS over WDM solution, so called λ-labeling, presented in [Gha00].

2.4 Wavelength Routed Networks

A wavelength routed network is an all-optical network, where the routing in the network nodes is based on the wavelength of the incoming signal [SB99]. The configuration of a WRN consists of choosing a free route and wavelength for each lightpath. Hence there are transparent optical channels, lighpaths, configured in the network. WRNs are very scalable and can achieve high utilisation degree in arbitrary mesh network.

All the subsequent chapters in this work concern WRNs. In Chapter 3 a static configuration problem is presented, and in Chapter 4 the lightpath requests follow some (stochastic) traffic process.

2.5 Linear Lightwave Networks

A linear lightwave network (LLN) is an optical network, where wavelength selective routing in the nodes is replaced by linear divider combiners (LDC) [BSSLB95, SB99]. A linear divider combiner is a linear network element, which combines signals from different sources in some proportion and then the combined signal emerges to given output ports in arbitrary fractions. Hence several wavelength channels are treated as a one unit[3], which often leads to so called fortuitous destinations [SB99]. LLNs are not considered in the rest of this thesis.

2.6 Logically Routed Networks

The possible layered approaches presented earlier lead to a concept of logically routed networks [SB99]. A logically routed network (LRN) is a concept where a logical topology is realised over a physical optical topology (see Fig. 2.5). Thus, transparent all-optical connections are configured into the physical network, and logical switching nodes (LSN) see the logical topology instead of the physical. This mapping allows changing the physical network, e.g. in case of some network failures, without changing the logical topology seen by the upper layers. Hence, it simplifies the description of the network to upper layers. A typical LSN could be an ATM switch.

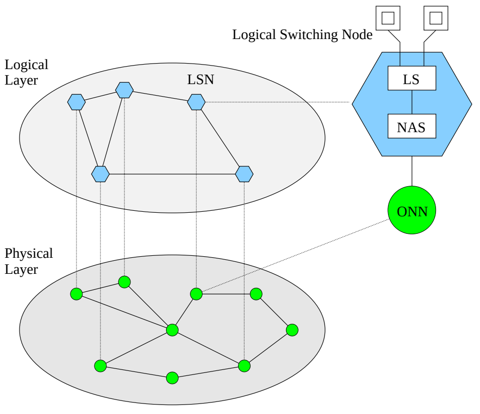

Figure 2.5: A logically Routed Network (LRN) where a logical topology is build over a physical topology. The Logical switching nodes (LSN) operate on logical topology, e.g. ATM or SONET switches (The figure is taken from [SB99]).

Hence, the connections end users require can be created between any (end user) node pair, i.e. there is a full connectivity. In the logical layer (ATM) switch finds a feasible route via possible zero or more intermediate LSNs between the LSNs the end users are connected to. Similarly, the optical layer supports the optical connections for the logical layer. The layered architecture of the optical networks is depicted in Fig. 2.6 [SB99].

[3] In waveband routing a certain set of channels, i.e. band.

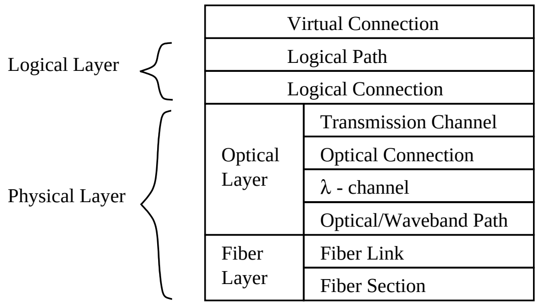

Figure 2.6: Layered architecture of the optical networks.

In practice the bandwidths the end users require are far below one optical channel. Thus, depending on the chosen logical topology the end user requirements can be fulfilled or not. The limitations on the logical topology are set by the physical network under it. Hence, in a generic level one can consider a problem where there are (time dependent) traffic flows between optical network nodes (ONN). These flows can be of any size from some proportion of a single optical channel to several channels. Once the logical topology is fixed the logical layer is suppose to route the data flows in it if only possible. This means that some data flows will be aggregated on the optical layer where they travel to the next LSN, which demultiplexes them to different data streams etc.

2.6.1 Multihop Network Configuration Problem

Generally, by a multihop network we mean such a network where the data flow (packet flow) requirements are no longer integer multiples of the capacity of a single wavelength channel, but arbitrary multiples or fractions of that capacity. Furthermore, these flows can be aggregated in any node to a single flow and later split again in some intermediate node, and then forwarded to other directions. Thus each data flow uses possibly more than one optical hop, hence the name. This causes extra processing load to the intermediate nodes and increases delays the packets experience, but allows more efficient use of the optical resources. The aggregation corresponds to the routing in the logical layer. Therefore, it is not usually practical to configure the network so that the logical and physical layers are topologically equivalent, because then the conversion between layers causes unnecessary delays to the traffic [RS96,MBRM96,ZA95].

This kind of problem can be solved in a hierarchical way presented in Fig. 2.7. In the first layer current (average) data streams between the nodes are mapped to lightpath requests, i.e. the requirements for the logical topology are set. If there is enough room in the network, each lightpath request can be fulfilled and a feasible solution is found.

Translation from the packet world to the lightpath world essentially defines the lightpath(s) each packet uses to travel through the network towards the destination node. Once this decision is made the problem reduces to assign-

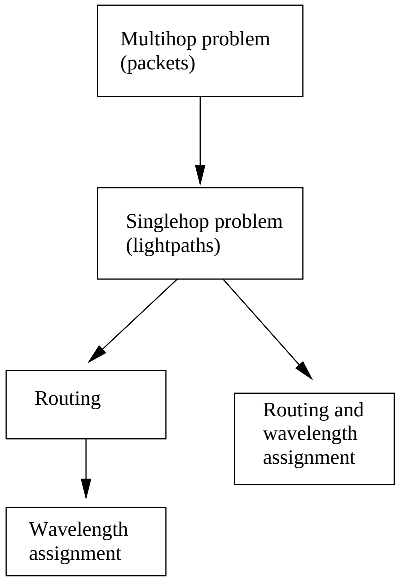

Figure 2.7: Hierarchical model of WDM network configuration.

ment of lightpaths in the network (the second box from the top in Fig. 2.7).

The establishment of lightpaths in the network, on the other hand, is traditionally solved in one or two phases. By one phase solution we mean an algorithm where both the route (for lightpath) and its wavelength(s) are determined simultaneously. By two phase solution we mean an algorithm where the path is first fixed for each lightpath and then a feasible wavelength is assigned to each lightpath (see Section 3.2). Generally shorter paths are usually good candidates.

In the later sections we concentrate on the configuration of the physical network layer, i.e. the problem is to configure the given lightpaths into the network. Or equivalently, a given logical topology is to be realized in the physical layer.

2.7 Optical Packet Switching

In contrast to the circuit switched WDM networks the optical packet switching offers even more flexibility. The often used idea is to build local area networks using optical packet switching. The photonic packet switching, however, involves many open questions [YMD00,PMMB00,BGD01]. The proposals can be divided to two categories, namely slotted and unslotted. In the slotted solution each packet has a constant length and the operation is synchronous, while in the unslotted case packet lengths can vary and the operation is asynchronous. Generally controlling the delay in photonic packet switched networks is an important issue.

In the electronic domain the contention in packet switching is resolved by a store-and-forward technique [YMD00]. Thus in the case of contention packets are stored in a queue from which they are sent further later. In the optical transmission the buffering is a complex task as there is no optical random access memory (RAM) available. The solution is to use fixed length delay lines.

Other possible schemes to deal with contention include deflection routing, where otherwise lost packets are sent to some other direction in the network, where they will be forwarded again towards the original destination.

The header format must be chosen carefully as the capacity of optical network is huge and the processing of the headers must be accomplished in a shorter time interval than what is the case in electrical networks. It is likely that header processing must be done first electronically which means a conversion from optical to electrical domain for routing decision, and later back to optical.

Chapter 3

Routing and Wavelength Assignment Under Static Traffic

3.1 Introduction

The routing and wavelength assignment (RWA) problem in wavelength routed WDM networks (WRN) consists of choosing a route and a wavelength (RW-pair) for each connection so that no two connections using the same wavelength share the same fibre [BM96, Bar98]. The requirement that connections sharing the same fibre must use different wavelength channels is often referred as *distinct channel assignment* requirement, or shortly DCA (see e.g. [SB99,RS98, Muk97]):

Definition 3.1 (Distinct Channel Assignment [DCA])
Connections sharing a common fibre must use distinct wavelengths.

A violation of the DCA constraint is often referred to as a *wavelength conflict*. Furthermore, if the wavelength conversion is not possible in the network nodes, then an additional constraint, called *wavelength continuity*, must be satisfied, i.e. each connection must use the same wavelength on every link. This constraint together with DCA give the RWA problem in all-optical network its special characteristics.

In this chapter it is assumed that traffic is static, i.e. the problem is to configure a given static set of connections between the given nodes into the network. The network itself can be a single or multifibre network. This kind of approach is relevant in the backbone networks, where it may be reasonable to assume that the traffic is static.

Another kind of problem formulation arises in the context of Linear Lightwave Networks (LLN) where wavelength selective routing in the nodes is replaced by Linear Divider Combiners (LDC) [BSSLB95, SB99]. This case is not considered in this thesis.

The static case of the routing and wavelength assignment problem has been widely studied in the literature. For the reference see e.g. [TP95,BB97,GSM97]. In this chapter we give a brief introduction to RWA problem under static traffic without going into any details.

Consider the following problem:

Problem 1 (Static Routing and Wavelength Assignment[S-RWA])

- *Nodes and links of physical network are given, where each link has a certain number of bidirectional fibres (i.e. fibre pairs)*
- *A static traffic matrix defining the number of lightpaths required between each node pair is given*
- *Wavelength conversion is not possible in the network nodes (usually)*
- **Problem**: *Determine a routing and wavelength assignment with minimal number of wavelengths*

The absence of wavelength conversion means that wavelength continuity constraint must be satisfied in addition to the DCA constraint. The current technology sets a bound to the maximum number of wavelengths available, and if the solution uses more than that it cannot be realized in practice. Hence, solving the problem answers the question whether a given traffic requirements can be satisfied with current physical network, or do we have to add new links (or fibres) to the network in order to be able to satisfy given traffic requirements. Later in Section 3.4 the problem definition will be extended to allow the installation of new fibres to the links with additional cost and the optimization goal becomes the minimization of the costs.

In Fig. 3.1 an example wavelength routed network is depicted. In the example network a lightpath is configured between each node pair. The wavelength translation is assumed to be impossible in the network nodes (WSXC). The shown configuration uses 5 wavelengths, which is the optimal configuration in this case.

3.2 Routing and Wavelength Assignment as Separate Problems

The static RWA problem can be solved either in one phase, where both route and wavelength is assigned at the same time, or in two phases, where first the routes are fixed and then a feasible wavelength assignment is determined for the given routing. The latter approach has proven to work quite well and is briefly explained in this section.

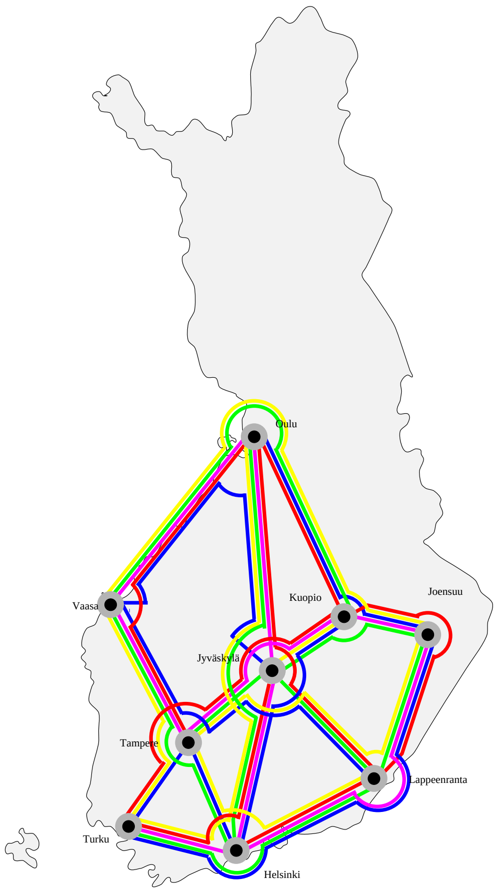

Figure 3.1: A hypothetical WDM network in Finland with a connection between every node pair.

3.2.1 Routing

The traditional way to solve the static RWA problem is to first determine the routes for all connections and then assign wavelengths to the connections. Even though the problems are not independent, this is likely to give moderately good configurations. The usual way to choose the routes is to choose (one of) the shortest path(s) for each connection (for shortest path algorithms see B.2). Longer routes use more network resources and are likely to lead to less efficient configurations. If there are several equally short paths, then one of them is randomly chosen. The optimal configuration is often obtained by using short routes, but not necessary the shortest one for every route (in order to avoid unnecessary congestion on some links).

3.2.2 Wavelength Assignment

The unique feature for WDM-networks is that wavelength conflicts are not allowed (DCA constraint), i.e. no two connections using the same wavelength may share a joint link (or fibre to be exact). Once routing is fixed the problem is to assign feasible wavelengths with minimum number of wavelengths in order to satisfy constraints set by the technology. In the general case there are several fibres between some of the links. A straightforward approach is to assign the lowest possible[1] wavelength to one connection at a time in some order (greedy algorithm). The order in which the wavelengths are assigned can be a critical factor for greedy algorithms and a good thumb rule is to assign a wavelength first to those connections which have most dependencies, i.e. share most links with other connections.

Algorithm 1 Static wavelength assignment algorithm

Let $\mathcal{K}$ be the set of all connections
$\forall k \in \mathcal{K}$, let $d_k = \sum_{i \in \mathcal{K}} I(i \text{ and } k \text{ use a common link})$
Sort connections in the decreasing order of d_k
Start from an empty network
for each $k \in \mathcal{K}$ in sorted order **do**
  Configure connection k into the network using the lowest possible wavelength(s)
end for

Graph Node Colouring Problem

The wavelength assignment in a single fibre network[2] is equivalent to the node colouring problem, which is an old and well-known graph theoretic problem (see e.g. [SK77, AH77, BM76]). In the node colouring problem the task is to assign a colour to each node of the given graph with minimal number of colours so that no neighbour nodes have the same colour (see B.5). Several heuristic algorithms have been applied to solve the problem including simulated annealing, tabu search and genetic algorithms [Ree95, RSORS96, HdW87].

The relation between the wavelength assignment and graph node colouring is the following. Assume that the set of routes is fixed and the task is to assign a feasible wavelength to each connection, i.e. no two connection sharing the same link may use the same wavelength. Let each connection represent a node in a graph and set such connections as neighbours which share at least one link. By finding the optimal colouring for this graph, we have also found the optimal wavelength assignment for the given routing. The bad news is that the graph node colouring problem is NP-complete. However, several reasonably well working heuristic algorithms exist (see e.g. [HdW87, Mit76, Bré79]).

[1]Such wavelength which does not cause a wavelength conflict
[2]Or in multifibre network if the routing step has fixed also the used fibre on every link.

3.2.3 Improving Current Solution

A straightforward way to improve the current solution is to change the set of routes a little and then assign the wavelengths again[3]. If the wavelength assignment is not computationally too expensive an operation, then well-known local search techniques like simulated annealing, genetic algorithms or tabu search can be applied to obtain the optimal set of routes (for a given wavelength assignment policy). Thus, the result from wavelength assignment is fed back to upper level algorithm as the value of the cost function in the current point (=routing). But as stated before, this requires a moderately fast wavelength assignment algorithm.

3.3 Some Bounds for the Number of Required Wavelengths

A lower bound for the number of wavelength channels required can be found by cutting the network into two parts [HV98,Bar98,SB99]. A certain number of connections cross this cut. By dividing the number of such connections by the number of physical fibres going through the cut we get the average number of connections using those fibres (assuming no connection crosses the cut more than once. Nonetheless, it is clear a lower bound for the number of connections crossing the cut). The minimum number of wavelengths must be greater than or equal to the average in order to avoid wavelength conflicts in those fibres, i.e.

$$W \geq \max_{\text{cut}} \frac{\text{number of connections through plane}}{\text{number of fibres crossing the plane}}.$$

An example network is presented in Fig. 3.2 where the optimal cut states that at least 3 wavelengths are required to realize a full connectivity within the example network.

In the case of a fully-connected network the number of connections crossing the cut is generally $N_A \cdot (N - N_A)$ where N is the total number of nodes and N_A is the number of nodes in part A. Hence, the maximum number of connections crossing a cut is obtained by dividing the nodes to two equally big groups, giving the total $\frac{N^2}{4}$ connections. However, the number of fibres crossing such a cut can be numerous and generally the tightest lower bound is obtained by going through all possible cuts. This is clearly infeasible in practice as the number of possible cuts can be enormously large. However, a heuristic algorithm can be used to find reasonably good cut within practical time.

On a single fibre link case when the problem has been reduced to graph node colouring problem (routes are fixed), the graph theory (see Appendix B) also

[3] The opposite, changing wavelengths little and then finding a feasible routing, is usually harder to solve.

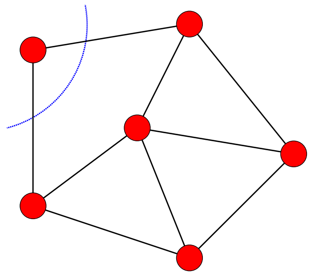

Figure 3.2: An example network with a cut. Assuming single fibre links and full logical connectivity this gives total 5 connections crossing the cut and hence at least $\lceil \frac{\mathbf{5}}{\mathbf{2}} \rceil = \mathbf{3}$ wavelengths are required.

gives a few bounds for the number of required wavelengths. For example, an upper bound for the number of required wavelengths is obtained from

$$W_{\min} \leq \Delta + 1,$$

where Δ is the maximum degree (the number of neighbours a connection has) of the connection graph. Graph theoretic bounds, however, are usually not very strict and thus not very useful in this context.

3.4 Network Planning

In the long run it is clear that new fibres have to be installed in the networks. In this section a formulation of how a network provider could plan its future network is presented. The model presented here assumes that some initial network exists and in order to meet the new (static) traffic requirements the network can be extended by adding new fibres to some links with certain costs.

Thus, it is assumed that a core network already contains a certain set of fibres, and with an extra cost new fibres can be installed between any node pair. The problem is to find the cheapest set of new fibres, so that the required connections can be configured in the network.

This problem formulation could be extended by allowing installation of WIXC in place of WSXC with an extra cost, leading to so called sparse wavelength conversion. The optimal location of the WIXCs is studied in [HMM99], but in this section the nodes are assumed to be fixed. The formulation will be extended to include the restoration requirements later in Section 6.6.

Problem 2 (Static Network Planning)

- *Nodes and links of physical network are given, where each link has a certain number of bidirectional fibres (i.e. fibre pairs)*

- *The number of available wavelength channels is given (same for every link)*
- *New fibres can be installed with extra costs*
- *A static traffic matrix is given, defining the number of lightpaths required between each node pair*
- **Problem***: Determine the cost-effective configuration for the network*

This problem definition is a generalization of the Problem 1 where addition of new fibres was not allowed. Here we will build the required logical connectivity in the optical domain no matter what it costs and the aim is to minimize the costs. Note that the type of the network nodes, WSXC or WIXC, is not fixed yet.

Cost from Fibre Installation

Let $\mathcal{N}$ be the set of network nodes and $\|\mathcal{N}\| = N$. Denote the set of all possible links by $\mathcal{L}$. The cardinality of $\mathcal{L}$ is clearly $N \cdot (N-1)$. The maximum of W wavelengths are available in each fibre, constraint the technology sets. Define the set of wavelengths as

$$\mathcal{W} = \{1, \ldots, W\}.$$

Let s_ℓ be the number of fibres already installed on link l. Formally, the cost of fibres on link ℓ is

$$R_\ell(m_\ell) = \text{the cost of } m_\ell \text{ fibres on link } \ell, \tag{3.1}$$

where the cost function is zero, when m_ℓ is smaller than or equal to the number of already installed fibres s_ℓ on link ℓ. Also, if it is practically impossible to install a fibre between some node pair, the cost function has some arbitrary high value for that link. Naturally, the form of the cost function depends on the case. Especially installing one fibre or several fibres at the same time usually costs about the same and the cost function is a monotonically growing step-function. But for the simplicity the following piecewise linear cost models can also be considered.

Assume, that the core network consists of single-fibre links ($s_\ell = 0$ or $s_\ell = 1$) and we are only interested in adding extra fibres to those links which have a fibre already, i.e. expanding their capacity. Assuming that installing a new fibre to any link has a constant cost, i.e. it does not depend on the length of the link etc., then the cost function could be defined as

$$R_\ell(m_\ell) = \begin{cases} 0, & \text{when } m_\ell \leq s_\ell, \\ m_\ell - s_\ell, & \text{when } s_\ell > 0, \\ \infty, & \text{otherwise}. \end{cases} \tag{3.2}$$

The presented model assumes a fixed installing cost for any link, where as in reality the costs can vary considerably. A more generic, but still partially linear model, which allows variation in link costs is the following:

$$R_\ell(m_\ell) = \begin{cases} 0 & \text{when } m_\ell \le s_\ell, \\ (m_\ell - s_\ell) \cdot R_\ell & \text{otherwise,} \end{cases} \tag{3.3}$$

where R_ℓ is a link specific cost factor for installing a new fibre to link ℓ. This model is clearly a bit more realistic than the previous one. The distance and other peculiarities of link can be taken into the account in some extent. Especially if it is infeasible to install additional fibres between some node pair, i.e. link ℓ, the cost factor $R_\ell = \infty$ (or some other arbitrary high value).

The cost function of the whole problem is the sum of the link costs $R_\ell(m_\ell)$ over all the links $\ell \in \mathcal{L}$:

$$\text{total cost} = E = \sum_{\ell \in \mathcal{L}} R_\ell(m_\ell), \tag{3.4}$$

which is to be minimized. There are no other sources of cost, i.e. the configuration of the network and such are assumed to be free.

Solving the Static Network Planning Problem

A simple approach to solve this more generic problem is to first solve the Problem 1 with current network without any additional fibres. If the solution uses too many wavelengths, then one fibre is added to one of the most congested links. Then the algorithm solving the Problem 1 is executed again. This is repeated until the number of wavelengths is equal to or lower than the set maximum.

Another way to approach the problem is to change routing and wavelength assignment as well as number of fibres at the same time. This will be considered in the following sections where first a formulation for the networks consisting of WSXC nodes is presented following a similar formulation for the networks with WIXC nodes.

3.4.1 WSXC Case

The nodes of network are assumed to be WSXC's, i.e. no wavelength conversion is possible. Thus the solution to the problem picks one path and wavelength for each connection. Let $\mathcal{A}_z$ be the set of possible paths for connection z. The model allowed addition of new fibres to the network, so $\mathcal{A}_z$ contains every path in a fully connected graph with the network nodes as vertices. Formally, the solution is a mapping,

$$f : \mathcal{Z} \to \mathcal{A}_z \times \mathcal{W},$$

i.e. for each $z \in \mathcal{Z}$, $f(z) = (p_z, c_z) = $ (path, colour). Of course the shortest path would be the one using the link directly connecting the source and target nodes. But if there is no fibre already installed between the nodes and the cost of installing such a fibre is huge, it is very probable that such a solution is not cost effective. Note that the cost realizing an arbitrary solution is defined by f, i.e. the solution f defines the amount of additional fibres, which in turn incurs costs.

The choice of the paths and wavelengths, f, defines the number of required fibres. If no new fibres are needed the cost is zero and we have found the optimal solution. For an arbitrary path p define $I(\ell \in p)$ to be 1 if path p goes through link l, and otherwise zero. Define a usage matrix $\mathbf{U}$ as follows.

$$u_{c,\ell} = \sum_{z \in \mathcal{Z}} I(\ell \in p_z, c = c_z),$$

i.e. $u_{c,\ell}$ is the number of connections using link ℓ with wavelength c. Thus, for each link ℓ the number of fibres required m_ℓ (multiplicity) is clearly

$$m_\ell = \max_{c \in \mathcal{W}} u_{c,\ell} = \max_{c \in \mathcal{W}} \sum_{z \in \mathcal{Z}} I(\ell \in p_z, c = c_z). \tag{3.5}$$

The total cost of given configuration f can be obtained by using (3.4). Note that we have not taken into account restoration aspects yet. These will be discussed in Section 6.6.

Pruning the Search Space

Assume that we have found one feasible solution f_0, which costs E units. Now we can limit the set of possible paths, i.e. prune the search space. If the minimum cost of using some path p is greater than E, then there is no need to consider that path. Formally if,

$$\sum_{\ell \in \mathcal{L}} I(\ell \in p) \cdot R_\ell(1) = \sum_{\ell \in p} R_\ell(1) \geq E, \tag{3.6}$$

then the path p cannot be part of better solution and can be excluded from $\mathcal{A}_z$. In particular, if for some link ℓ it holds that $R_\ell(1) \geq E$, then that link and all the paths using it can be excluded from the search space. Denote the pruned subset of $\mathcal{A}_z$ with $\mathcal{A}_{z,E}$. The pruning basically removes infeasible links from the search space and also prunes individual paths as well. Further pruning of the solution space (mappings) with some heuristic algorithms is probably necessary as well to achieve reasonable running times.

3.4.2 WIXC Case

If all the nodes of the network have capability of wavelength conversion, the formulation becomes slightly different, since with wavelength translation capable nodes each connection can switch to another wavelength on the next link. Hence, the wavelength continuity constraint is no longer valid and the solution f only defines the used route for each connection. The total number of connections using the same link defines the number of required fibres. Formally, the solution f is

$$f : \mathcal{Z} \to \mathcal{A}_z,$$

i.e. for each connection $z \in \mathcal{Z}$, the solution f is $f(z) = (p_z) = (\text{path})$. Also the multiplicity on link ℓ is simply

$$m_\ell = \left\lceil \frac{1}{W} \sum_{z \in \mathcal{Z}} I(\ell \in p_z) \right\rceil ,$$

i.e. the total number of users is divided by the number of available wavelengths W rounded up to the nearest integer. Again, once the multiplicity m_ℓ is known for every link ℓ the total costs can be calculated using (3.4). The similar pruning technique as was used in the WSXC case can be applied also here.

3.4.3 Implementation of Local Search

The routing and wavelength assignment problem is clearly very hard, so heuristic algorithms are the only possible alternative. Here it is assumed that some kind of local search method is used to find a feasible solution. Small changes are made to the configuration hoping that it will eventually converge to the global minimum.

The crucial points in every local search heuristic is how the neighbourhood and energy function are defined. A straightforward choice for the energy function is the cost of installing new fibres to the network. It is, however, probably not the best choice as wide areas in the search space give constant value for the total cost and local search method has no information about the direction to proceed.

Two definitions for neighbourhood are given below.

Neighbourhood I

This neighbourhood of the current solution f consists of such functions f^* where the path of one connection is changed. All the wavelengths can be different (WSXC case). Denote the rerouted connection with z_r. Then, the mappings $f = (p,\ c)$ and $f^* = (p^*,\ c^*)$ are neighbours if for some z_r[4]

$$\begin{cases} p^*(z) &= p(z), \quad \forall\, z \neq z_r \\ p^*(z) &\neq p(z), \quad \text{when } z = z_r \end{cases}$$

[4] Actually z_r partitions the set of neighbours of f.

In practice, after some connection is rerouted all the wavelengths are reassigned while trying to minimize the maximum number of connections using the same wavelength on any link which defines the multiplicity for each link.

Neighbourhood II

This neighbourhood is much smaller than the previous one. The neighbourhood of the current solution f consists of such functions f^* where only one connection z_r has either new path, new wavelength or both. All the other connections remain the same, i.e. wavelengths are not reassigned. Formally, the mappings f and f^* are neighbours if for some z_r it holds that

$$\begin{cases} f^*(z) & = & f(z), \quad \forall\, z \neq z_r \\ f^*(z) & \neq & f(z), \quad \text{when } z = z_r \end{cases}$$

Thus, f and f^* are neighbours iff exactly one connection uses different route or wavelength (or both). This neighbourhood seems to be too small for fast convergence.

Relationship between neighbours: WIXC

An efficient way to speed up an algorithm is to use previous values when calculating the new one.

In case of WIXC nodes the neighbourhoods I and II are identical. One must keep book only about the number of users on every link, i.e. updating vector $\mathbf{U}_L$ accordingly. The usage vector for a neighbour solution can be obtained by first decrementing the usage on the corresponding links along the old path, and then incrementing usage on corresponding links along the new path.

The number of fibres needed per link is the number of connections divided by the number of wavelength rounded up:

$$m_\ell = \left\lfloor \frac{u_\ell}{W} \right\rfloor .$$

Relationship between neighbours: WSXC

If the network consists of WSXC nodes then the network usage is defined by the matrix $\mathbf{U}_{W\times L}$ (instead of a vector). The number of fibres needed per link is simply the maximum value of column ℓ:

$$m_\ell = \max_c u_{c,\ell},$$

which defines the value of the cost function.

In the neighbourhood I the wavelengths can be reassigned after changing the route of one connection. In wavelength reassignment only the column sums of the matrix $\mathbf{U}$ are preserved. Thus, a similar speeding up is not possible in this case.

On the other hand with the neighbourhood II only some of the matrix $\mathbf{U}$ entries change when the system moves to a neighbour state (i.e. from one solution to another, f to f^*). Hence there is no need to recalculate the whole $\mathbf{U}$ matrix but instead a new usage matrix can be obtained recursively from the previous. Assume that connection z_r is rerouted. Then the procedure of updating $\mathbf{U}$ and obtaining the cost of the new solution becomes

1. Update the usage matrix $\mathbf{U}$:
   - Remove the old connection $f(z_r) = (p(z_r),\ c(z_r))$ is from $\mathbf{U}$, i.e. subtract a binary vector $p(z_r)$ from the $c(z_r)^{th}$ row of $\mathbf{U}$
   - Add a new connection $f^*(z_r) = (p^*(z_r),\ c^*_(z_r))$, i.e. add a binary vector $p^*(z_r)$ to $c^*(z_r)^{th}$ row of $\mathbf{U}$
2. Update the number of the fibres required on each link m_ℓ:
   - the number of required fibres m_ℓ can be lower only if $\ell \in p(z_r)$ and $m_\ell = u_{c(z_r),\ell}$ before removal
   - the number of required fibres m_ℓ increases by one ($m_\ell \leftarrow m_\ell + 1$), only if $\ell \in p^*(z_r)$ and $m_\ell = u_{c^*(z_r),\ell}$ before removal
3. Determine the value of the cost function (3.4), which is a function of the number of fibres m_ℓ

3.5 Summary

In this chapter the configuration problem of static WDM-networks was briefly formulated. Two alternative formulations were presented; Problem 1 answers the question whether given connections can be configured to the current physical optical network, while in problem 2 the goal is to minimize the cost from installing new fibres to satisfy given traffic requirements.

The optical layer is used to form optical connections between the given set of nodes. Above the optical layer the higher layers will support the required connectivity. The static traffic case is quite reasonable an assumption for the backbone networks, where new connections arrive very rarely. The static routing and wavelength assignment problem (S-RWA) is an NP-complete problem and only heuristic algorithms are practical.

Chapter 4

Routing and Wavelength Assignment Under Dynamic Traffic

4.1 Introduction

In this chapter it is assumed that traffic is not static but lightpath requests arrive randomly following some traffic process. Hence, the routing and wavelength assignment constitutes a typical decision making problem. When a certain event occurs, one has to decide on some action. The set of possible actions is finite: either reject the call or accept it and assign a feasible combination of route and wavelength (RW) to it. A feasible RW combination is such that along the route from the source to destination the wavelength is not being used on any of the links. If no feasible RW combination exists, the call is unconditionally rejected. Furthermore, the accepted connections cannot be interrupted.

As in the previous chapter, we consider wavelength routed networks (WRN). The network nodes are assumed to be WSXC's, hence without the capability to do wavelength conversions.

The dynamic routing and wavelength assignment problem (D-RWA) is studied mainly in the setting of Markov Decision Processes (MDP). The application of MDP theory in the context of routing problems is not new. For example, Krishnan and others have applied the MDP theory with traditional circuit switched networks [Kri90, Kri91, ZAA$^+$97, Rum00, Ahl00]. The same problem arises in the context of WDM networks as with traditional circuit switched networks, i.e. the astronomical size of the state space makes it impossible to solve the optimal policy in most cases. Hence, more or less heuristic algorithms are the only option.

The schemes considered under dynamic traffic can be divided into two cases: reconfigurable and non-reconfigurable. If it is possible to reconfigure the whole network when blocking would occur, the blocking probability can be considerably reduced. Such an operation, however, interrupts all (or at least many) active lightpaths and requires a lot of coordination between all the nodes. In large networks the reconfiguration seems infeasible. In any case, the reconfiguration algorithm should try to minimize the number of reconfigured lightpaths in order to minimize the amount of interruptions in the service [MM99].

The other case occurs when active lightpaths may not be reconfigured. In this case it is important to decide which route and wavelength are assigned to an arriving connection request in order to balance the load and minimize the future congestion in the network.

In general, one is interested in the optimal policy which maximizes or minimizes the expectation (infinite time horizon) of a given objective function. Here we assume that the objective is defined in terms of maximizing or minimizing some revenue or cost function. The cost may represent e.g. the loss of revenue due to blocked calls, where different revenue may be associated to each type of call.

Problem 3 (Dynamic Routing and Wavelength Assignment [D-RWA])

- *Nodes and links of physical network are given, where each link has certain number of bidirectional fibres (i.e. fibre pairs)*
- *Lightpath requests (bidirectional connections) arrive according to some stochastic model*
- *Active connections yield revenue, or alternatively the blocked connection requests generate costs*
- **Problem**: *Dynamically control the network (CAC) so that the revenues are maximized, or alternatively the costs are minimized*

Connection requests between a given source destination pair constitute a *traffic class*, which is indexed by k, $k \in \mathcal{K}$, where $\mathcal{K}$ is the set of all source destination pairs. When the arrival process of class-k calls is a Poisson process with intensity λ_k, the holding times of those calls are distributed exponentially with mean $1/\mu_k$, and the revenue rate per active class-k connection is w_k, then the system constitutes a Markov Process and the problem of determining the optimal policy belongs to the class of Markov Decision Processes (MDP) described e.g. in [Tij94] and [Dzi97]. Appendix C contains a brief introduction to the subject.

A *policy*, usually denoted with α, defines for each possible state of the system and for each traffic class k of an arriving call which of the possible actions is taken. Usually the RWA algorithm configures lightpaths in the network unless there are not enough resources available and the request is blocked, i.e. unconditionally accepts requests whenever possible. This is, however, not always the optimal policy.

Three main approaches for solving the optimal policy in the MDP setting are the policy iteration, value iteration and linear programming approach [Tij94, Tah92]. In Section 4.2 the policy iteration is applied to obtain the optimal policy to a simple case with three network nodes. The reason for the small size of the example network is the prohibitive size of the state space for any more realistic network sizes, which would make it impossible to solve exactly the optimal policy.

Many heuristic policies have been proposed in the literature such as the first-fit wavelength and most-used wavelength policies combined with shortest path routing or near shortest path routing, see e.g. in [KA98, MA98, RS95]. Some of them work reasonably well. In [SB97] a wavelength assignment algorithm for a fixed routing is presented. Common to all these heuristic policies is that they are rather simple. The choice of the action to be taken at each decision epoch can usually be described in simple terms and does not require much computation. These algorithms, however, do not take into account the traffic characteristics like unequal costs of different requests or inhomogeneous arrival rates. In Section 4.3 some of the heuristic algorithms are described.

Section 4.4 contains a brief survey of approximate methods to estimate the blocking probability of given WDM network.

In Section 4.5, we concentrate again on the policy iteration, where, as the name says, one tries to find the optimal policy by starting from some policy and iteratively improving it (see e.g. [Tij94, Dzi97]). The policy iteration is known to converge rather quickly to the optimal policy. Even the first iteration often yields a policy which is rather close to the optimal one. In practice, it is seldom possible to go beyond the first iteration[1]. Also in this work, we will restrict ourselves to the first policy iteration. In order to avoid dealing with the huge size of the state space in calculating the relative state costs needed in the policy improvement step, we suggest to estimate these costs on the fly by simulations for the limited set of states that are relevant at any given decision epoch, i.e. when the route and wavelength assignment for an arriving call has to be made. The first policy iteration approach in the context of WDM-networks was introduced in [HV00a] and then studied in more detail in [HV00b].

4.2 Exact Calculation of the Optimal Policy

In this section we shall show how the optimal call admission policy can be calculated for an arbitrary network. The simple network in Fig. 4.1(a) is used to demonstrate the steps in the calculation and the prohibitive size of the general problem. The links in the network are assumed to be bidirectional single fibre links. The lightpath requests are as well bidirectional, an assumption made in

[1] An exception is the case where we have the perfect information about the future presented in Section 4.7

later sections as well. We assume the wavelength continuity constraint here, i.e. there is no wavelength conversion capability in the network nodes. In the example network each connection between any node pair can be routed in exactly two ways, directly or via the third node. We refer to these routes as the primary (P) and the secondary (S) route. As will be seen later, these calculations, however, are not feasible even for a moderate size network, not to speak about a large network.

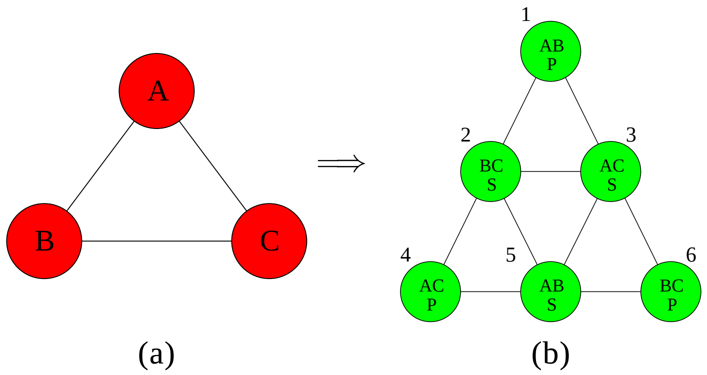

Figure 4.1: A simple three node network (a) and possible routes between the nodes (b). Routes are neighbours if they cannot be active at the same time.

The graph in Fig. 4.1(b) represents the relations between possible routes. The routes are set as neighbours (connected by an edge) if they share a common link. Each time a new connection is configured into the network essentially one node is taken in use (coloured). The purpose of the connection graph is to define the allowed configurations. Connections having an edge between them in connection graph may not be set up concurrently because that would cause a wavelength conflict (DCA constraint, see Def. 3.1). Thus, a configuration is feasible if and only if the active connections form an independent set[2] of (multi layer) graph of Fig. 4.1(b).

Each wavelength constitutes an identical layer in Fig. 4.1(a). As the wavelength conversion is not allowed in any node, the lightpath must travel using only one layer. In case of a WIXC node connections could jump from one layer to another in those nodes. The lack of wavelength conversion means also that the route neighbour graph (Fig. 4.1(b)) becomes a multiple layer graph with no edges between the layers as depicted in Fig. 4.2 for the two wavelength layers case. As subgraphs consisting of one wavelength layer are identical, it is enough to consider one layer and then generalize the results.

As an example connection graph see Fig. 4.2. As stated before, denote with $\mathcal{K}$ the set of traffic classes, i.e. in the example case simply $\mathcal{K} = \{AB, AC, BC\}$. A traffic class consists of bidirectional connections between the given nodes. Each traffic class has a unique arrival rate λ_k, a unique service rate μ_k and a unique revenue rate w_k per time unit. The model can be generalized to include several traffic classes between the same pair of nodes having different characteristics like higher revenue per time unit. Assuming the arrival process to

[2] An independent set is set of vertices which have no edge between them.

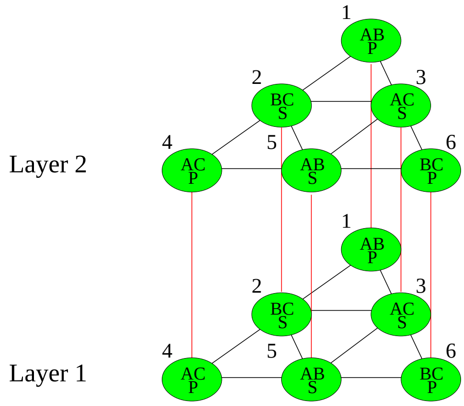

Figure 4.2: Connection graph with two wavelength layers ($\boldsymbol{W = 2}$). The layers have no edges between them (the vertical red edges in the figure represent that the nodes use the same route but different wavelength, and can be in use at the same time).

be Poisson and service times to obey the exponential distribution the dynamic routing and wavelength assignment constitutes a traditional Markov Decision Process (MDP), which is briefly presented in Appendix C.

4.2.1 Obtaining the Set of Possible States $\mathcal{S}$

Assume we are given a labelled graph $\mathcal{G}$, which defines the possible routes in the network. Denote the set of vertices (or nodes) by $\mathcal{V}$. Thus, each vertex v has three attributes, which define (a) the traffic class k, $k \in \mathcal{K}$, (b) the route and (c) the wavelength, i.e. if vertex v is active, then a class-k connection is configured to network using certain route and certain wavelength. In case of no wavelength conversion the connection graph $\mathcal{G}$ consists of W identical and independent layers of $\mathcal{G}_0$, i.e.

$$\mathcal{G} = \mathcal{G}_0 \oplus \mathcal{G}_0 \oplus \ldots \oplus \mathcal{G}_0.$$

As stated before, the connections must be configured to the network so that no wavelength is used more than once in any single link in order to avoid the wavelength conflicts. This means that the set of possible states in which the network can be, consists of those subsets of graph $\mathcal{G}$ vertices which have no edges between them, i.e. which form an independent set. In case of no wavelength conversion this is equivalent to finding all the independent sets of graph $\mathcal{G}_0$ and taking all the possible W combinations of them, i.e.

$$\mathcal{S} = \mathcal{S}_0 \times \mathcal{S}_0 \times \ldots \times \mathcal{S}_0 = \mathcal{S}_0^W,$$

where $\mathcal{S}_0$ is the set of independent sets of graph $\mathcal{G}_0$, $\mathcal{S}$ is the set of independent sets of graph $\mathcal{G}$ and W is the number of wavelengths available. Once the independent sets $\mathcal{S}_0$ of graph $\mathcal{G}_0$ are obtained, it is straightforward to form all the possible combinations and renumber the states. Hence, the size of the state space can easily be very large leading to infeasible calculations, as will be shown in the following paragraphs. In Section 4.2.4 a method to reduce the size of the state space with the aid of the symmetry of the wavelength layer will be presented.

We get an equivalent formulation if we consider the complement graph $\bar{\mathcal{G}}$ of graph $\mathcal{G}$. In a complement graph the set of vertices is the same, but there is an edge between nodes u and v if and only if there is no such edge in the original graph. Then a feasible configuration is characterized by the fact that the routes in use (=vertices of graph) must form a clique, i.e. each pair of vertices has an edge between them. Hence, the state space of active connections is equivalent to the set of all cliques of graph $\bar{\mathcal{G}}$.

Well-known algorithms for generating all the cliques/independent sets of an arbitrary graph exist (see e.g. [KS99, Öst99, AJ83]). The problem itself is NP-complete and thus intractable for other but small graphs (see Appendix B.3).

In the example graph $\mathcal{G}_0$ of Fig. 4.1 the set of independent sets $\mathcal{S}_0$ is

$$\begin{aligned}\mathcal{S}_0 = \{ \quad & \emptyset, \\ & \{1\},\{2\},\{3\},\{4\},\{5\},\{6\}, \\ & \{1,4\},\{1,6\},\{4,6\},\{1,5\},\{2,6\},\{3,4\} \\ & \{1,4,6\} \qquad\qquad\qquad\qquad\qquad\qquad\qquad \}\end{aligned}$$

There are in total 14 possible states for each wavelength. Hence, the total number of states in the example network is 14^W. For $W = 4$ this gives 38416 states, which means that the transition probability matrix $\mathbf{Q}$ has about $1.5 \cdot 10^9$ entries. Assuming 32 bit floating point arithmetic, storing the matrix $\mathbf{Q}$ requires about 6 Gbytes of memory.

4.2.2 Routing and Wavelength Assignment Policy

The policy α defines the route and wavelength for an incoming connection request based on the current state of the network. Formally

$$\alpha : \mathcal{S} \times \mathcal{K} \to \mathcal{S},$$

with the following constraint

$$\alpha(i,k) = i \ \vee \ \alpha(i,k) = i \cup \{v\} \text{ where } L(v) = k.$$

The first case represents blocking, i.e. the state of the network does not change. In the other case the incoming call is accepted. The function L is a (label) mapping from the set of vertices $\mathcal{V}$ to traffic classes $\mathcal{K}$:

$$L : \mathcal{G} \to \mathcal{K},$$

i.e. it returns the traffic class associated with the given node. The constraint accounts for that the added lightpath v fulfills the class-k request. Denote the set of possible new states when a class-k arrival occurs in state i with $\mathcal{A}_{i,k}$. Then it holds that $\alpha(i,k) \in \mathcal{A}_{i,k}$.

4.2.3 Transition Probability Matrix Q and Revenue Rates

The goal of stochastic optimization is to maximize the expected revenue (or equivalently to minimize the expected costs). For this, we need to define the revenue rate for each state of the system (or equivalently revenues could be associated pointwise with transitions). Assume that in each state i the system accrues revenues in certain fixed rate r_i. So during a certain period of time T spent in state i the total revenues would be $r_i \cdot T$.

Once the policy α is fixed the transition probability matrix $\mathbf{Q}(\alpha)$ can be written down. After the steady-state probabilities $\boldsymbol{\pi}(\alpha)$ of the system, defined by $\mathbf{Q}(\alpha)$, are solved the average revenue rate of the system can be obtained,

$$r(\alpha) = \sum_{i \in \mathcal{S}} r_i(\alpha)\pi_i(\alpha),$$

where $r_i(\alpha)$ is the revenue rate in state i under policy α. The optimal policy α is the policy which maximizes the average revenue rate $r(\alpha)$.

Assume that each active connection generates incomes at a constant rate w_k which depends on the traffic class k. Then it is clear that the revenue rate in state i is independent of the used policy α as the policy only determines call admission, i.e. no matter what policy is used the revenue rate in state i depends only on the currently active connections. Thus, the revenue rate in state i is the sum of active connections weighted with the appropriate traffic class weighs,

$$r_i = \sum_{v \in i} w_{L(v)}, \tag{4.1}$$

where w_k defines the revenue per time unit accrued by a class-k connection in progress.

Another equivalent way to phrase the revenue rates is to look at the expected loss of revenue due to the missed requests, i.e.

$$r_i(\alpha) = \sum_{k \in \mathcal{K}, \alpha(i,k)=i} -\beta_k \cdot \lambda_k, \tag{4.2}$$

where $\beta_k = w_k/\mu_k$ is the expected total revenue per a carried class-k connection[3]. The negative revenue rate corresponds to the fact that we are actually looking at the cost rates instead of revenue.

These two forms for the revenue rate have the following relation. For each traffic class $k \in \mathcal{K}$ it holds that,

$$\underbrace{B_k(\alpha)\lambda_k\beta_k}_{\text{cost rate}} + \underbrace{\overline{N}_k(\alpha)w_k}_{\text{income rate}} = \underbrace{\lambda_k\beta_k,}_{\text{offered income rate}}$$

[3]Then, however, the cost from the blocked call is not included in the relative costs v_i used in Howard's equations, and must be taken into account explicitly when policy iteration is applied.

Listing 4.1: A Matlab function to calculate $\mathbf{Q}_\mu$ and $\mathbf{r}$.

```
function [Qmu, r] = generate_Qmu_and_r(S,L,mu,w );
%
% Calculate Qmu and r. Qmu has all (constant) departure rates
% and r likewise constant revenue rates from each state.
%
% S is a NS*NZ matrix, where NS is number of states and NZ maximum
% number of members in a state (set). State is (v1,v2,..,vx) i.e.
% a set of nodes in graph G.
% L is mapping from G to K, where K is traffic classes.
% mu and w are K-vectors containing departure and revenue rates.

NS  = size( S, 1 );
Qmu = zeros( NS, NS );
r   = zeros( NS, 1 );

for s1=1:NS;
  s = S{s1};
  for ss=1:length(s);
    v=s(ss);                     % active node?
    if v > 0                     % an active call
      k  = L(v);                 % traffic class
      ns = setxor( s, v );       % new state after departure
      s2 = get_state( S, ns );
      Qmu(s1,s2) = Qmu(s1,s2) + mu(k);
      r(s1)      = r(s1) + w(k);
    end;
  end;
end;
```

where $B_k(\alpha)$ is the blocking probability of the class-k connection and $\overline{N}_k(\alpha)$ is the average number of active class-k connections when the used policy is α. The formula is a direct consequence of Little's result $N = \lambda T$. By taking a sum over all the traffic classes the total income rate $r(\alpha)$, the total cost rate $c(\alpha)$ and the total offered income rate can be obtained. Hence,

$$r(\alpha) + c(\alpha) = \sum_k \lambda_k \beta_k = \text{constant},$$

and

$$\arg\max_\alpha \; r(\alpha) = \arg\min_\alpha \; c(\alpha).$$

The departures from the system are independent of the used policy. But transitions due arrivals are policy dependent. Thus, the transition intensity matrix $\mathbf{Q}(\alpha)$ can be decomposed into two parts,

$$\mathbf{Q}(\alpha) = \mathbf{Q}_\mu + \mathbf{Q}_\lambda(\alpha), \tag{4.3}$$

where matrix $\mathbf{Q}_\mu$ is policy independent part containing the departures and matrix $\mathbf{Q}_\lambda(\alpha)$ contains the arrivals and depends on the used policy. This decomposition is useful when the transition probability matrix $\mathbf{Q}(\alpha)$ is needed to obtain consequently to new policies, as is the case with the policy iteration.

Listing 4.1 contains a Matlab function which obtains the policy independent parts, i.e. the departure rates defined in the matrix $\mathbf{Q}_\mu$ and the revenue rate vector $\mathbf{r}$. The revenue rates are written according to definition (4.1) so that $\mathbf{r}$ does not depend on the chosen policy α. Thus, the revenue vector $\mathbf{r}$ contains simply the sum of income rates of the active connections in each state.

Listing 4.2: A Matlab function to calculate $\mathbf{Q}$ for any fixed policy $\boldsymbol{\alpha}$.

```
function Q=generate_Q ( Qmu, lambda, alpha );

Q    = Qmu;                                   % constant departure rates
Snum = size( Qmu, 1 );
Knum = length( lambda );

for s = 1:Snum;                               % Snum possible states
    for k = 1:Knum;                           % Knum traffic classes
        ss = alpha(k,s);                      % new state
        if ss ~= s                            % call accepted
            Q(s,ss) = Q(s,ss) + lambda(k);
        end;
    end;
    Q(s,s) = 0;
    Q(s,s) = -sum( Q(s,:) );
end;
```

The Matlab function presented in Listing 4.2 then produces the transition rate matrix $\mathbf{Q}(\alpha)$ for given policy α by adding the policy dependent arrival rates to $\mathbf{Q}_\mu$ according to equation (4.3). Basically, the algorithm goes through every possible state and arrival, and updates the matrix $\mathbf{Q}(\alpha)$ accordingly. Then the algorithm updates the entries on the diagonal so that the row sums of the matrix $\mathbf{Q}(\alpha)$ are equal to zero (as required by definition of $\mathbf{Q}$, see C.2).

Continuous Time Howard's equations

Now, instead of solving the steady-state probabilities for each possible policy α and maximizing $r(\alpha)$ with respect to α, a better way exists to find the optimal policy, namely using the continuous time Howard's equations and applying the policy iteration (see Appendix C.5). By solving the continuous time Howard's equations

$$r_i(\alpha) - r(\alpha) + \sum_{j \neq i} q_{ij}(\alpha) \cdot (v_j(\alpha) - v_i(\alpha)) = 0, \qquad \forall\, i \in \mathcal{S},$$

we get the so called relative values of states i, $v_i(\alpha)$, and the average revenue rate $r(\alpha)$, under policy α. Because $q_{ii} = -\sum_{j \neq i} q_{ij}$ for all i, Howard's equations can also be written as

$$r_i(\alpha) - r(\alpha) + \sum_{j} q_{ij}(\alpha) \cdot v_j(\alpha) = 0, \qquad \forall\, i \in \mathcal{S}.$$

One of the relative values can be given an arbitrary value, e.g. we can set $v_0(\alpha) = 0$. Then the set of linear equations can be written in the familiar form,

$$\begin{pmatrix} 1 & -q_{1,2} & -q_{1,3} & \dots & -q_{1,n} \\ 1 & -q_{2,2} & -q_{2,3} & \dots & -q_{2,n} \\ \vdots & & \ddots & & \vdots \\ 1 & -q_{n,2} & -q_{n,3} & \dots & -q_{n,n} \end{pmatrix} \cdot \begin{pmatrix} r \\ v_2 \\ \vdots \\ v_n \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{pmatrix},$$

which is an equation of the form $\mathbf{A}\mathbf{x} = \mathbf{b}$, where $\mathbf{x}$ is an unknown vector. It has the formal solution

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b},$$

but the inverse of a huge matrix is infeasible to calculate. If $n \times n$ matrix $\mathbf{A}$ is sparse and moderate size then the equation can be solved using, for example, some numerical package.

Policy Iteration Step

The possible decisions in state i when a class k arrival occurs are

a) configure the connection into the network using a feasible RW pair, or

b) reject it.

The state of the system defines the active connections (and how they are configured), and as stated before, the set $\mathcal{A}_{i,k}$ contains all the possible states the system can be immediately after the decision for a class-k arrival in state i is made. The state after the decision can be a new state (transition) or the same state in the case the connection request is blocked. Furthermore, this post decision state defines the decision made uniquely, i.e. policy α defines a new state j, $j \in \mathcal{A}_{i,k}$, for each $i \in \mathcal{S}$ and $k \in \mathcal{K}$.

Once the relative values v_i and average revenue rate r are obtained for the current policy α, the policy iteration technique is applied to improve the current policy (see Appendix C.6). Assume that the objective function describes the revenue rates in each state, i.e. we want to maximize the average revenue rate. Recall, that the relative value v_i defines the expected future revenues when the system starts from state i using the given policy α.

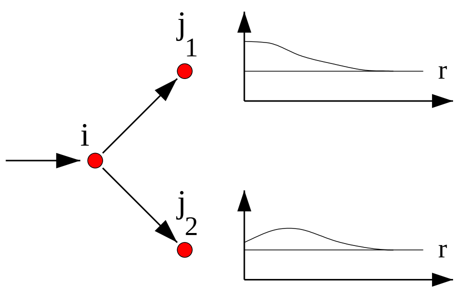

Figure 4.3: Two possible decisions depicted. Decision j_1 corresponds to a case where incoming call is accepted while in j_2 it is rejected (revenue rate is lower at $t = 0$).

In this case the rejection itself costs immediately nothing but accepting a connection gives no immediate revenue either. Hence, there is no revenues involved immediately to the decision, but the revenue (or loss of it) is included in the future evolution of the system, i.e. it is included in the relative values v_i.

It makes sense to choose such a post arrival state that has highest expected (relative) revenues among all possible states $\mathcal{A}_{i,k}$. Thus, the policy iteration formula simply becomes,

$$\alpha'(i,k) = \underset{j \in \mathcal{A}_{i,k}}{\arg\max} \{v_j(\alpha)\}, \qquad \forall\, i,k.$$

The equation defines the action to be taken in each state i for any possible arrival k, i.e. a new policy α', which can be shown to be never worse than the original policy α [Tij94]. The new policy α' defines a new transition rate matrix $\mathbf{Q}$, and the policy iteration step can then be carried out for it in turn. This is repeated until the optimal policy is obtained, i.e. the average revenue rate $r(\alpha)$ does not improve further.

Policy Iteration with Lost Calls

If we concentrate on the lost calls, the formulation becomes a little bit different. Instead of maximizing the revenue rate, consider minimizing the average cost rate. Similarly, as defined in (4.2), the cost rate in state i is

$$r_i(\alpha) = \sum_{k:\alpha(i,k)=i} \lambda_k \cdot \beta_k,$$

i.e. the sum of the arrival rates of the traffic classes which are blocked in given state multiplied with appropriate average loss of revenues β_k per call.

Also the effect of blocking a connection request in policy iteration must be taken into account explicitly, as it is no longer included in the relative values v_i (the less active connections there are initially, the less blockings will occur on average!). Hence, the policy iteration step becomes

$$\alpha'(i,k) = \underset{j \in \mathcal{A}_{i,k}}{\arg\min} \{1_{i=j} \cdot \beta_k + v_j(\alpha)\}, \qquad \forall\, i,k. \tag{4.4}$$

In the above equation the cost from blocking the current connection request, i.e. if $j = i$, is explicitly added as it is not included in the relative costs v_j

4.2.4 Wavelength Degeneration

In this section a method to reduce the size of the state space $\mathcal{S}$ is presented. Because the wavelength layers are identical, the optimal policy should also exhibit the same symmetry. Hence, exchanging wavelengths λ_1 and λ_2 should have no effect on the optimal policy. Remember that the state space $\mathcal{S}$ is actually a Cartesian product of W layers. Thus, states s_1 and s_2 can be considered equivalent iff

$$\exists \text{ wavelength permutation } \pi : \mathcal{S}_0^W \to \mathcal{S}_0^W \text{ such that } \pi(s_1) = s_2,$$

where $\mathcal{S}_0$ is the independent set of one layer ($\mathcal{S}_0^W = \mathcal{S}$) and π is some permutation which exchanges the order of the wavelength layers.

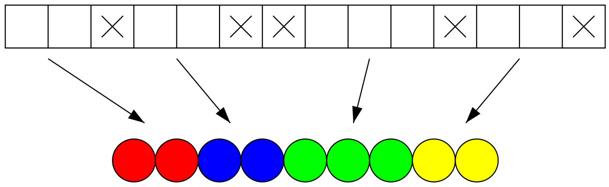

Figure 4.4: The number of ways to colour n balls with m colours. There are $n+m-1$ slots and marking $m-1$ slots divides the rest n slots to m groups.

Next we will determine the number of possible states when the symmetrical states are removed. Assume that there are N states in each layer, i.e. $N = ||\mathcal{S}_0||$. The states can be enumerated with numbers from 1 to N.

Depict the wavelength layers as “balls” and the state of each layer as a “colour” of the ball. Then an equivalent formulation is to find out the number of ways to assign colours to W identical balls from the set of N colours without any restriction to how many times each colour should be used. Note that the balls are not numbered and only the number of balls having the same colour matters.

Colouring n identical balls with m colours

In this section a classical combinatorial formula is briefly studied. Assume n identical balls (order of balls does not matter) is to be given a colour using m different colours. Let $X(n,m)$ be the number of combinations to colour n identical balls with m colours, where $n \geq m$, so that each colour is given to at least one ball. Similarly, let $Y(n,m)$ be the number of combinations to colour the balls without the requirement to use each colour at least once. This is equivalent to the Bose-Einstein statistics used in physics, where the particles of the system are assumed to be unidentifiable. The balls are the particles and the colour of the ball corresponds to the energy state of the particle etc.

Consider that there are $n+m$ balls and m colours. Then after each colour is given to one ball there are still n balls left. Thus it holds that

$$Y(n,m) = X(n+m,m) \qquad \forall\, n,m \geq 1.$$

Clearly, if there is only one colour available, i.e. $m=1$, the balls can be coloured only in one way, and hence $X(n,1) = Y(n,1) = 1$. Similarly, if the number of colours is equal to the number of balls, i.e. $m = n$, then only one possible colour assignment using all the colours exists, i.e. $X(n,n) = 1$ for all $n \geq 1$. Furthermore it is easy to see that e.g. $X(n,2) = n-1$.

The general formula for $Y(n,m)$ can be easily obtained. Consider placing $n+m-1$ slots in a row (see Fig. 4.4). By marking $m-1$ of them the rest n slots (=balls) are partioned to m (possibly empty) sets where the first set represents the balls given the 1^{st} colour etc. Hence there is a one-to-one relation between the different colourings of n balls and markings of $m-1$ slots, i.e. the number of colourings is equal to the number of ways to choose the slots. The marked slots can be chosen in $\binom{n+m-1}{m-1}$ different ways, and thus

$$Y(n,m) = \binom{n+m-1}{m-1} = \binom{n+m-1}{n}, \qquad \text{where } n,m \geq 1. \tag{4.5}$$

From above it also follows that

$$X(n,m) = Y(n-m,m) = \binom{n-1}{n-m}, \qquad \text{where } n \geq m.$$

Also the following recursive equation[4] holds for $X(n,m)$:

$$X(n,m) = \sum_{i=1}^{\min\{m,n-m\}} \binom{m}{i} \cdot X(n-m,i),$$

or alternatively

$$X(n+m,m) = \sum_{i=1}^{\min\{m,n\}} \binom{m}{i} \cdot X(n,i).$$

Number of States in the Reduced State Space

As was explained before the total number of states in the reduced state space is equal to the number of ways to assign colours to W balls using N colours without any restriction to how many times each colour should be used. Applying the formula (4.5) gives that the total number of states after removal of the degenerated states is

$$||\mathcal{S}_d|| = Y(W,N) = \binom{N+W-1}{W}. \tag{4.6}$$

While this reduces the size of the state space quite a lot, from N^W to $\binom{N+W-1}{W}$, the general problem remains still intractable.

For the simple example network of Fig. 4.1 with $N = 14$ and $W = 4$, the formula gives $||\mathcal{S}_d|| = 2380$, while the number of states before removal of equivalent states was 38416.

4.2.5 Relationship between $\mathcal{S}$ and $\mathcal{S}_d$

Let s $= \begin{pmatrix} s_1 & s_2 & \ldots & s_W \end{pmatrix}$ be a state in the original state space $\mathcal{S}$. Then define that s belongs to $\mathcal{S}_d$ iff $s_1 \geq s_2 \geq \ldots \geq s_W$. That is the states included in the

[4] each i includes the cases where i colours are used more than once.

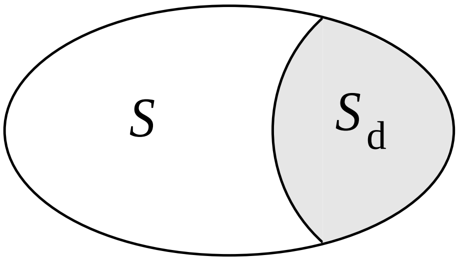

Figure 4.5: The original and degenerated state space.

reduced state space are those whose elements in vector representation are in decreasing order. It is trivial to find the mapping from $\mathcal{S}$ to $\mathcal{S}_d$ by sorting the vector in decreasing order (sorting a vector is equal to taking certain permutation).

In the reduced state space there is one element representing each equivalence class. Denote by $\{\pi_s\}_{s\in\mathcal{S}}$ the family of permutations which exchange the order of wavelength layers so that

$$\forall\ s \in \mathcal{S} : \pi_s(s) \in \mathcal{S}_d.$$

The permutation π_s is a bijection and thus has a trivial inverse function π_s^{-1} which exchanges the wavelengths back to the original order. Furthermore, for all $s \in \mathcal{S}_d$, it holds that $\pi_s(s) = s$. Let f be a surjective mapping from $\mathcal{S}$ to $\mathcal{S}_d$ defined by the permutations above:

$$f : \mathcal{S} \to \mathcal{S}_d, \qquad f(s) = \pi_s(s).$$

Hence,

$$\mathcal{S}_d = \{s_d : \exists\ s \in \mathcal{S}, f(s) = s_d\},$$

or in other words,

$$\mathcal{S}_d = \{s : f(s) = s\}.$$

The policy $\alpha^{(d)}$ in $\mathcal{S}_d$ is feasible if there exists a policy α in $\mathcal{S}$ for which it holds

$$\alpha^{(d)} = f \circ \alpha.$$

Denote by $\mathcal{A}_{s,k} \subset \mathcal{S}$ the set of possible new states when class-k arrival occurs in state $s \in \mathcal{S}$. Similarly, $\mathcal{A}^{(d)}_{s_d,k} \subset \mathcal{S}_d$ is the set of possible new states for $s_d \in \mathcal{S}_d$. Note that the policy essentially chooses one state from each $\mathcal{A}_{s_d,k}$ or $\mathcal{A}^{(d)}_{s_d,k}$. Generally, the policy α defines a transition rate matrix $\mathbf{Q}$ and a revenue rate vector $\mathbf{r}$. The reduction of the state space also gives us new $\mathbf{Q}_d$ and $\mathbf{r}_d$ in a natural way. Note that the revenue rate from a connection does not depend on the used wavelength (or even on the route). Thus, the states where the wavelengths are just in different order will have the same revenue rates.

Once the sets $\mathcal{A}^{(d)}_{s_d,k}$ are found for all k and s_d, the policy iteration can be used to obtain the optimal policy in $\mathcal{S}_d$. Then the optimal, but not necessarily unique, policy in $\mathcal{S}$ is the one for which

$$f(\alpha(s,k)) = \alpha^{(d)}(f(s), k).$$

Generally, for an arbitrary state $s \in \mathcal{S}$ and a connection request $k \in \mathcal{K}$, the optimal policy is

1. if $s \in \mathcal{S}_d$, then $\alpha(s,k) = \alpha^{(d)}(s,k)$,
2. if $s \notin \mathcal{S}_d$, then $\alpha(s,k) = \pi_s^{-1}(\alpha^{(d)}(\pi_s(s),k)$,

which is well-defined as $\pi_s(x)$ is a bijection.[5]

The whole procedure to obtain the optimal policy is presented in Algorithm 2.

Algorithm 2 Optimal policy algorithm

Enumerate possible routes for each traffic class
Determine graph $\mathcal{G}$ which holds dependencies between routes, i.e. whether they share a link or not
Determine the state space $\mathcal{S}$, i.e. independent sets of graph $\mathcal{G}$
Form the reduced state space $\mathcal{S}_d$
Form the initial policy $\alpha_0^{(d)} : \mathcal{S}_d \times \mathcal{K} \to \mathcal{S}_d$
repeat
  Use policy iteration step to find a better policy $\alpha_{i+1}^{(d)}$
until no improvement on average revenue rate
Form the optimal policy α from $\alpha^{(d)}$ (optimal policy in reduced state)

4.2.6 Worst Case Scenario

In the worst case scenario the graph $\mathcal{G}_0$ has no edges. Assuming there are V nodes in it, this gives in total 2^V possible independent sets, i.e. states in which the system can be. Let there be K traffic classes, each having R alternative routes with W possible wavelengths, and assume that there are no wavelength conflicts. Then we get

$$||\mathcal{S}_0|| = 2^{KR},$$

where $\mathcal{S}_0$ is the independent set of one layer of graph $\mathcal{G}$. Hence, the state space of any moderate size network is likely to be prohibitively large. Moreover, enumeration of all independent sets of graph $\mathcal{G}_0$ is not always possible in practice, as the maximum independent set problem is a well-known NP-complete problem and by enumerating all the independent sets we also find the maximum independent set.

By combining the wavelength degeneration analysis (formula (4.6)) and the worst case scenario for the state space, the size of the state space in worst case becomes

$$||\mathcal{S}_d|| = \frac{N \cdot (N-1) \cdot \ldots \cdot (N+W-1)}{W!}, \text{ where } N = 2^{KR}.$$

[5]Actually the mapping $\alpha(s,k) = \pi_s^{-1}(\alpha^{(d)}(\pi_s(s),k)$ would be enough as $\pi_s(s) = s$ when $s \in \mathcal{S}_d$.

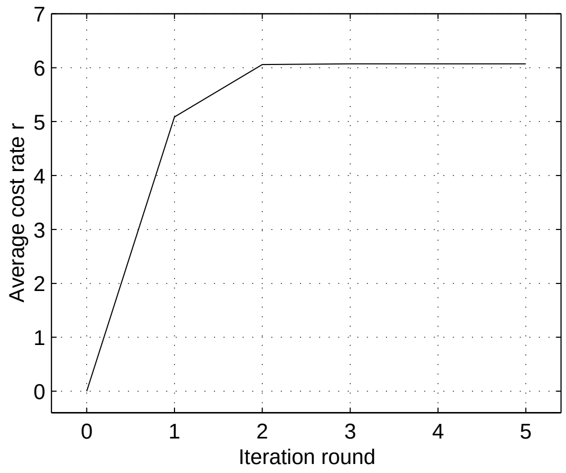

Figure 4.6: The evolution of the average cost rate r as a function of iteration rounds. The startup policy was "reject-all". Already after first two iteration rounds the policy is almost optimal.

The size of the state space grows extremely fast!

4.2.7 Triangle Network Example

In this section the optimal policy is calculated for the example network of Fig. 4.1 with one and two wavelength layers ($W = 1, 2$). The used traffic parameters are presented in Table 4.1. The parameter x defines the weight of class AB connections and it is varied between $0.0 - 5.0$ in order to see the critical points where the optimal policy changes. Clearly, when $x = 0$ the traffic between nodes A and B is considered worthless and should be rejected always. Likewise, when x becomes large enough the traffic class AB should be the only one accepted (as it uses every link in the system). As the network is symmetric and the traffic classes are homogeneous when $x = 1$ the optimal policy should be equal to all the traffic classes at that point and the blocking probability curves are equal at point $x = 1$ and possibly in its neighbourhood.

	λ	μ	w
AB	1.0	1.0	x
AC	1.0	1.0	1.0
BC	1.0	1.0	1.0

Table 4.1: Traffic parameters for each class.

In Fig. 4.6 the evolution of the policy iteration is illustrated. The x-axis represents the iteration round and in the y-axis is the average revenue rate. In this figure the number of wavelength layers is $W = 2$ and the weight factor $x = 5$. The algorithm finds the optimal policy in 5 iteration rounds starting from the worst possible policy, i.e. rejecting every connection request. In the next step most of the connection requests are accepted, and after the first two rounds the

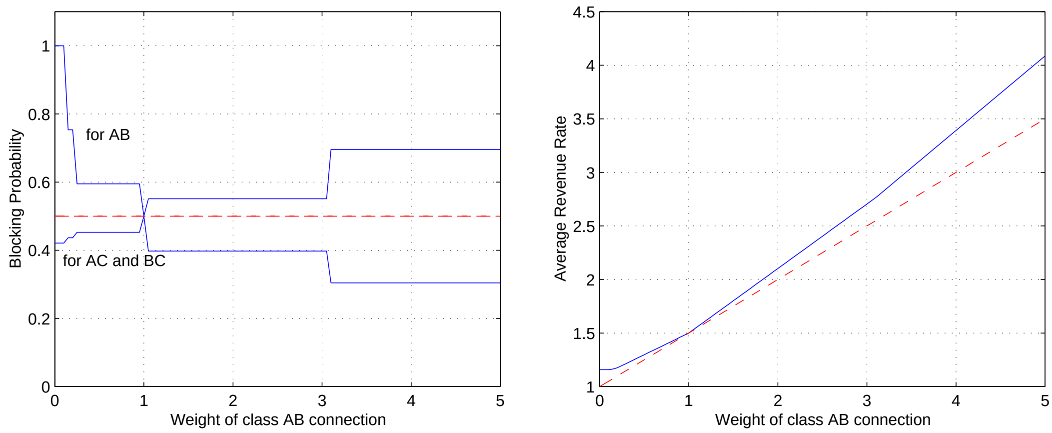

Figure 4.7: The evolution of the blocking probability and the average revenue rate with the optimal policy compared with a simple fixed policy. The network has one wavelength layer ($\boldsymbol{W} = \mathbf{1}$).

average revenue rate is almost equal with the optimal policy. The initial policy for the policy iteration is the worst possible, and any greedy RWA policy (see Section 4.3) is likely to give near optimal policy with one iteration round. This kind of behaviour where the solution converges near the optimal in a few steps is very advantageous for the first policy iteration to be presented in Section 4.5. Typically the number of iteration rounds needed with $W = 1, 2$ is from 5 to 10 rounds with these parameters and this network.

One Wavelength Layer

In Fig. 4.7 the optimal policy is illustrated with one wavelength layer ($W = 1$). The straight dashed lines in both the blocking and the average revenue rate figures represent the results with a fixed first-fit algorithm (see Section 4.3). The weight factor of class AB connection varies between 0 and 5. When $x = 0$ the class AB is worthless and should never be accepted, which indeed is the case (curve starting from $(0, 1)$). The average revenue rate increases approximately linearly with x. Every time the blocking probabilities change the optimal policy α changes too. After about 1.0 the optimal policy remains the same up to around 3.1. It is clear that with high enough x the optimal policy will be a policy which accepts only class AB connections. But with $x = 5$ still about 30% of the lower value AC and BC connections are accepted.

Two Wavelength Layers

In Fig. 4.8 the number of wavelength layers is two ($W = 2$), but otherwise the situation is the same as before. As the capacity of the network is increased the blocking probabilities of each traffic class will decrease and the average revenue rate will increase. It can be noticed that the area where all traffic classes

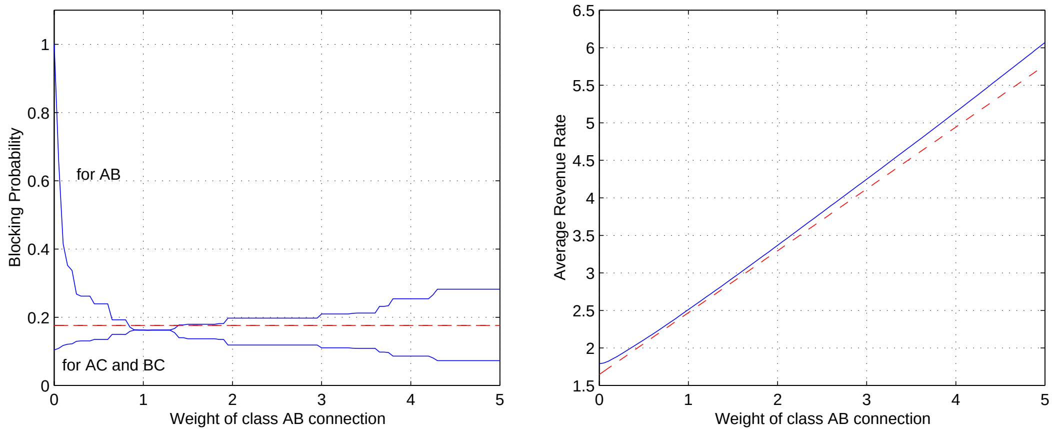

Figure 4.8: The evolution of the blocking probability and the average revenue rate with the optimal policy compared with a simple fixed policy. The network has two wavelength layers ($\boldsymbol{W = 2}$).

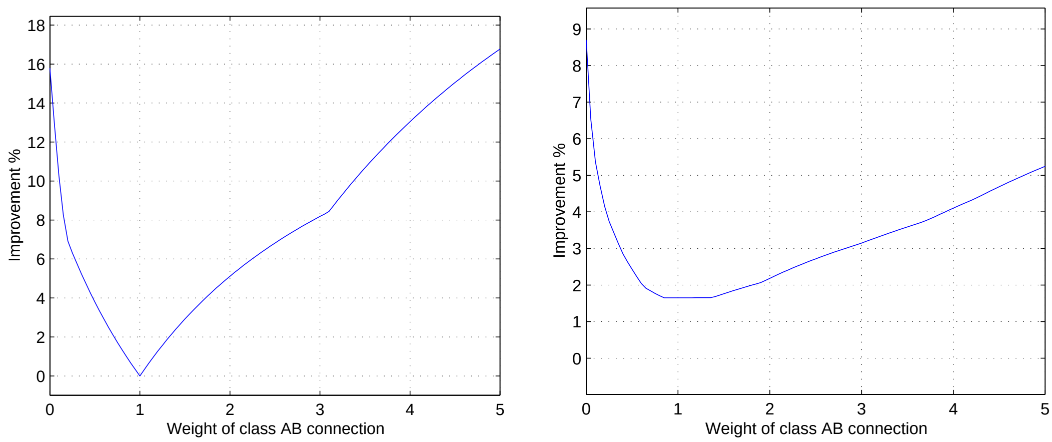

Figure 4.9: The improvement over the simple first-fit policy in one and two wavelength layers.

become equal policy wise has become wider around the point $x = 1$. This is an expected result since the capacity of the network has doubled and blockings have become more rare, so it makes sense to accept more connections. Also in the two wavelength layer case the optimal policy gives always a higher average revenue rate than the simple first-fit policy.

Comparing the average revenue rates with $W = 1$ and $W = 2$ we notice that the proportional improvement over the fixed first-fit policy is smaller with the two wavelength layers case. This can be also seen from Fig. 4.9 where the improvement over the simple first-fit algorithm is depicted as a function of class AB weight x. This suggests that with the smaller blocking probabilities and (near) homogeneous traffic the simple first-fit policy works quite well. Asymptotically both curves converge to a constant value because the optimal policy converges to one which rejects all other traffic classes except AB. In the limit the optimal policy has average revenue rate $(1 - B_{AB}) \cdot x$, while the simple first-fit policy has the average revenue rate $(1-B') \cdot x$ (the blocking is the same for every traffic class). Hence, the proportional improvement converges to $(B' - B_{AB})/B'$.

4.3 Heuristic Algorithms

Several quick heuristic algorithms for the dynamic RWA problem have been proposed in the literature. Here we briefly present some of them and later in Chapter 5 study how the first iteration approach works with them. The first set of algorithms assumes that a fixed set of possible routes for each connection is given in advance. Some papers refer to this as an alternate routing strategy. In practice the set of routes usually consists of the shortest or nearly shortest paths. These algorithms are greedy and accept the first feasible RW pair they find (first-fit).

- *basic* algorithm goes through all the routes in a fixed order and for each route tries all the wavelengths in a fixed order. The routes are sorted in the shortest route first order. The new connection is routed on the first path on which a wavelength channel is available. Among the available wavelength channels the first feasible channel is selected (see e.g. [KA96, RS95]).
- *porder* algorithm is similar to *basic*-algorithm, but it goes through all the wavelengths in a fixed order and for each wavelength tries all the routes in a fixed order.
- *pcolor* algorithm works like *porder*, but wavelengths are searched in the order of their current usage instead of a fixed order, so that the most used wavelength is tried first.
- *lpcolor* algorithm is the “smartest” algorithm. It packs colours, but the primary target is to minimize the number of used links. The algorithm first tries the most used wavelength with all the shortest routes, then the next often used wavelength and so on. If no wavelength works, the set of routes is expanded to include routes having one link more and wavelengths are tried again in the same order.

The above heuristics in slightly different forms are presented e.g. in [MA98, SB97, KA98].

Another set of heuristic algorithms, adaptive unconstrained routing (AUR) algorithms, are described in [MA98]. These algorithms search a route based on the current state of the network (dynamic routing) instead of relying on a fixed set of routes, and are thus a little bit slower.

- *aurpack* is similar to *pcolor*, but without the limitations of a fixed set of routes, i.e. routes of any length are acceptable.
- *aurexhaustive* finds a shortest route for each wavelength (if possible) and chooses the shortest RW pair among them, i.e. it is identical to *lpcolor* except that the set of routes is not fixed.

Thus AUR-algorithms will search for a free route dynamically based on the current state of the network. There is no need to store possible routes (which without any limitations can form a very large set) in advance.

Also other heuristics are given in [MA98], e.g. *random* (tries wavelengths in random order) and *spread* (tries least used wavelength first), but they were reported to work worse than the ones described above, and are not further discussed here.

4.4 Approximative Numerical Methods

As has been shown in Section 4.2, solving D-RWA problem exactly is not possible for any practical size mesh network. Hence, the blocking probabilities of connection requests can only be approximated. This can be done by running simulations and evaluating the blocking probability from them, or by using some approximate models of the system which then can be solved exactly or numerically. In this section a brief introduction is given on the latter approach.

A simple model, presented in [BH96, SB99], assumes *mutual independence* between all the links and all the wavelengths. Furthermore a fixed routing is assumed, i.e. one route is fixed for every source-destination pair (instead of alternate routing), and that the blocking probability is equal in all the links and wavelengths. Let ρ be the probability that a wavelength is in use on a link.

Then in the case of no wavelength translation (WSXC) the probability that a connection request is blocked becomes

$$P_b = \left(1 - (1-\rho)^H\right)^W , \tag{4.7}$$

where H is the number of links in the route and W is the number of wavelengths available. The formula can be explained in the following way. First, $(1-\rho)^H$ is probability that a route of length H is free on a fixed wavelength. Thus, $1-(1-\rho)^H$ is the probability of its complement. As the wavelength layers were assumed to behave identically and independently, it follows that the probability that none of the wavelength layers has a free route is $(1-(1-\rho)^H)^W$.

Similarly, in the case of WIXC, i.e. when wavelength translation is possible in every node, the blocking probability of a connection request is

$$P_b = 1 - \left(1 - \rho^W\right)^H . \tag{4.8}$$

In this case ρ^W is the probability that no wavelength channel is free on some link. Links behave identically and independently, and thus $(1-\rho^W)^H$ is the probability that there is one free wavelength channel available in every link along H link long route. Hence, $1-(1-\rho^W)^H$ is the probability that at least one of the links is full and connection request is blocked.

The assumption of the independence is valid when the network is dense and the number of wavelength channels is high. Then every link has many active connections which meet only in one link. More accurate models for estimating the blocking probabilities are presented e.g. in [RRP99, ZRP98a, ZRP98b, Bir96, SB97].

4.5 First Policy Iteration

In Section 4.2 the D-RWA problem was handled in the context of MDP theory. First Howard's equations were solved for the current policy and then a method called policy iteration was used to obtain a better policy α'. This was repeated until the optimal policy was obtained and the average revenue rate no longer improved.

As observed in Section 4.2.1 the size of the state space of any realistic size network is astronomical, and though Howard's equations are just a set of linear equations for relative costs v_i and the average cost rate $r(\alpha)$ of the standard policy, their solution cannot be obtained. Hence, it is practically impossible to determine the optimal policy for any realistic size network and other solutions must be sought.

In Section 4.3 several heuristic algorithms were presented. A deficiency in all the presented heuristic algorithms is that they do not take into account the possible additional information about the arrival rates, the distribution of holding times, or the priorities of traffic classes (different costs / revenues). Also the duration of the call when it arrives could be known (for example one channel is reserved for a certain event which lasts exactly two days), which conflicts slightly with the original assumptions about the traffic process (memoryless property). We could of course try to come up with better heuristics which somehow take into account the additional information, but that means that we would need a new heuristic policy for each new case.

The algorithm presented in this section still relies on the MDP theory. We take one of the heuristic policies presented in Section 4.3 as a starting point and call it the *standard policy*. Then the first round of the policy iteration is taken to make the actual decision. The policy resulting from the first policy iteration is referred to as the *iteration policy*. As stated before, it is not possible to solve all the relative values v_i due to the prohibitive size of the state space. However, at any decision epoch the relative values v_i are needed only for the small set of states $\mathcal{A}_{i,k}$ reachable from the current state (linear function of the number of traffic classes) when class-k arrival has occurred. In [HV00a] we propose *to estimate these values on the fly by means of simulations.*

Briefly, our idea in the first policy iteration is the following: at each decision epoch we make a decision analysis of all the alternative actions. For each of the possible actions, i.e. decision alternatives, we estimate the future costs by simulation. Thus, assuming that a given action is taken we let the system proceed from the state where it is after that action and use the standard policy to make all subsequent decisions in the simulation. The iteration policy is the policy which is obtained when at each decision epoch the action is chosen for which the estimated cost is the minimum. It can be shown that the iteration policy is always better or at least as good a policy as the standard policy, and as said, it often comes rather close to the optimal policy (see Fig. 4.6). The procedure to obtain the iteration policy, i.e. the policy resulting from the first policy iteration, is presented in Algorithm 3.

By doing the first policy iteration we have two goals in mind. 1) Finding a better D-RWA algorithm which, being computationally intensive, may or may not be calculable in real time, depending on the time scale of the dynamics of the system. 2) Even in the case the algorithm is not calculable in real time, estimating how far the performance of a heuristic algorithm is from the optimal one.

The iteration policy automatically adapts to the new situation, because the simulation, as explained later, will automatically take into account all the peculiarities of the system. So, even if the first iteration approach is very simple, it is very powerful due to its flexibility.

4.5.1 Relative Costs of States

In the MDP theory, the first policy iteration consists of the following steps: With the standard policy one solves the Howard's equations (see, e.g. [Tij94, Dzi97]) to obtain the so called relative costs of the states, v_i, which for each possible state i of the system describe the difference in the expected cumulative cost from time 0 to infinity, given that the system starts from state i rather than from the equilibrium. Assume that we are considering costs from the blocked connection requests instead of revenues from accepted connections (see Section 4.2.3). Then, given that the current state of the system is i and a class-k call is offered, one calculates the cost $\beta_k + v_i$, where $\beta_k = w_k/\mu_k$, for the action that the call is rejected, and the cost v_j, $j \in \mathcal{A}_{i,k}$ and $j \neq i$, for the case the call is accepted. The set $\mathcal{A}_{i,k}$ is the set of possible states after decision when the current state is i and call-k connection is assigned a feasible RW pair or rejected. By choosing always the action which minimizes the cost, one gets the iteration policy, i.e. the policy resulting from the first policy iteration.

Given that the system starts from state i at time 0, i.e. $X_0 = i$, and the standard policy α is applied for all the decisions, the cumulative costs are accrued at the expected rate $c_t(i)$ at time t,

$$c_t(i) = \mathrm{E}[r_{X_t} | X_0 = i] = \sum_k \lambda_k \beta_k \mathrm{P}\{X_t \in \mathcal{B}_k | X_0 = i\}, \tag{4.9}$$

i.e. the expected rate of lost revenue, where β_k is average revenue of carried class-k connection and $\mathrm{P}\{X_t \in \mathcal{B}_k\}$ is the probability that at time t the state X_t of the system is a blocking state for class-k calls under the standard policy. When $X_t \in \mathcal{B}_k$, class-k calls arriving at time t are blocked by the standard policy, because either no feasible RW pair exists or the policy otherwise deems the blocking to be advantageous in the long run. The expected cost rate $c_t(i)$ depends on the initial state i. However, no matter what the initial state is, as t tends to infinity, the expected cost rate tends to a constant r, which is specific to the standard policy, and corresponds to (4.9) with steady state blocking probabilities $\mathrm{P}\{X_t \in \mathcal{B}_k\}$.

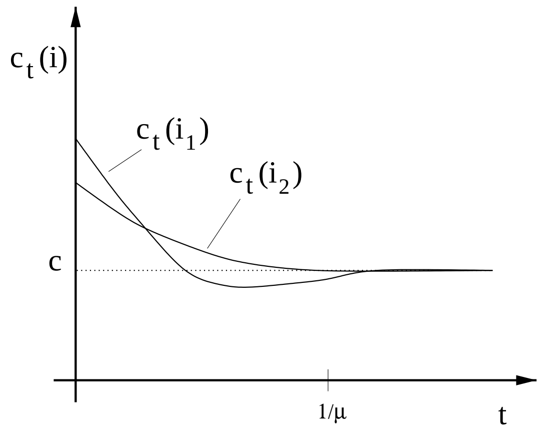

Figure 4.10: Expected costs with different initial choices as a function of time.

The behaviour of the function $c_t(i)$ is depicted in Fig. 4.10 for two different initial values i_1 and i_2. The relative cost v_i is defined as the integral

$$v_i = \int_0^\infty (c_t(i) - r)\, dt,$$

i.e. the area between the curve $c_t(i)$ and the line at level r. So we are interested in the transient behaviour of $c_t(i)$; after the transient no contribution comes to integral. The length of the transient is of the order $1/\mu$, where $1/\mu$ is the maximum over $\{1/\mu_k\}$, $k \in \mathcal{K}$. After this time the system essentially forgets the information about the initial state. So we can restrict ourselves to an appropriately chosen finite interval $(0, T)$. The actual choice of T is a tradeoff between different considerations as will be discussed later.

One easily sees that in the policy improvement step (see C.6) only the differences of the values v_i between different states are important. Therefore, we can neglect the constant r in the integral, as it is common to all states, and end up for thus redefined v_i,

$$v_i \approx v_i(T) = \int_0^T c_t(i)\, dt, \tag{4.10}$$

which is simply the expected cumulative cost in interval $(0, T)$ starting from the initial state i.

4.5.2 Estimation of the Relative Costs by Simulation

In practice, it is not feasible to calculate the cost rate function $c_t(i)$ analytically even for the simplest policies. Therefore, we estimate the relative costs v_i by simulations. In each simulation the system is initially set in state i and then the evolution of the system is followed for the period of length T, making all the RWA decisions according to the standard policy.

Statistics Collection: Blocking Time vs. Blocking Events

In collecting the statistics one has two alternatives. Either one records the time intervals when the system is in a blocking state of class-k calls, for all $k \in \mathcal{K}$. If the cumulative time within interval $(0, T)$ when the system is in the blocking state of class-k calls is denoted by $\tau_k(i)$, then the integral is simply

$$\hat{v}_i = \sum \lambda_k \beta_k \tau_k(i). \tag{4.11}$$

Alternatively, one records the number $\nu_k(i)$ of blocked calls of type k in interval $(0, T)$. Then we have

$$\hat{v}_i = \sum \beta_k \nu_k(i). \tag{4.12}$$

In these equations we have written explicitly $\tau_k(i)$ and $\nu_k(i)$ in order to emphasize that the system starts from the state i. Both (4.11) and (4.12) give an unbiased estimate for $v_i(T)$. In either case, the simulation has to be repeated a number of times in order to get an estimator with small enough a confidence interval.

Denote the estimates of future costs obtained in the jth simulation replication by $\hat{v}_i^{(j)}$, using (4.11) or (4.12) as the case may be. Then our final estimator for v_i is

$$\hat{v}_i = \frac{1}{N} \sum_{j=1}^{N} \hat{v}_i^{(j)}, \tag{4.13}$$

where N is the number of simulation replications. In fact, for the policy improvement the interesting quantity is the difference

$$E_{i_1,i_2} = v_{i_2} - v_{i_1},$$

for which we have the obvious estimate

$$\hat{E}_{i_1,i_2} = \hat{v}_{i_2} - \hat{v}_{i_1}. \tag{4.14}$$

From the samples $\hat{v}_{i_1}^{(j)}$ and $\hat{v}_{i_2}^{(j)}$, $j = 1, \ldots, N$, we can also derive an estimate for the variance $\hat{\sigma}^2_{i_1,i_2}$ of the estimator $\hat{E}_{i_1,i_2}$

$$\hat{\sigma}^2_{i_1,i_2} = \frac{N \sum_j (\hat{v}_{i_2}^{(j)} - \hat{v}_{i_1}^{(j)})^2 - \left(\sum_j \hat{v}_{i_2}^{(j)} - \hat{v}_{i_1}^{(j)} \right)^2}{N^2(N-1)} = \frac{\hat{S}^2_{i_1,i_2} - (\hat{E}_{i_1,i_2})^2}{N-1},$$

where $\hat{S}^2_{i_1,i_2} = \frac{1}{N}\sum_j \left(\hat{v}^{(j)}_{i_2} - \hat{v}^{(j)}_{i_1}\right)^2$.

The choice between the alternative statistics collection methods is based on technical considerations. Though estimator (4.11) (blocking time) has a lower variance per one simulation replication, it requires much more bookkeeping and the variance obtained with a given amount of computational effort may be lower for estimator (4.12) (blocking events).

Policy Iteration with Uncertain Relative Costs

The important parameters of the simulation are the length of the simulation period T and the number of simulation replications N used for the estimation of each v_i. In practice, we are interested in the smallest possible values of T and N in order to minimize the simulation time. However, making T and N too small increases the simulation noise, i.e. error in the estimates $\hat{v}_i$, occasionally leading to decisions that differ from that of the true iteration policy, consequently deteriorating the performance of the resulting algorithm.

No matter how the parameters are selected, some uncertainty in the estimators $\hat{v}_i$ is unavoidable. In order to deal with this uncertainty of the estimators $\hat{v}_i$, we do not blindly accept the action with the smallest estimated cost, but give a special status for the decision which would be chosen by the standard policy. Let us index this action with i_0. Based on the simulations we form estimates $\hat{E}_{i_0,i}$ for each possible action a.

Eq. (4.4) in Section 4.2.3 defines the policy iteration step for state i and arrival k,

$$\alpha'(i,k) = \underset{j \in \mathcal{A}_{i,k}}{\arg\min} \left\{ 1_{i=j} \cdot \beta_k + v_j(\alpha) \right\}, \qquad \forall\, i,k.$$

where the term $1_{i=j} \cdot \beta_k$ corresponds to the blocking and is so called immeadiate cost of the action $i \to j$. There is *no uncertainty* involved in the immeadiate costs. Immediate cost is either zero if the connection is accepted, or β_k if the connection is blocked. When two alternative actions leading to states i_1 and i_2 are compared, only the difference in immediate costs, i.e.

$$H_{i_1,i_2} = \beta_k \cdot (1_{i_2=i} - 1_{i_1=i})$$

is important.

Note that the information about if either action blocked the connection request is included in the destination states i_1 and i_2 (one need not know i). The difference in the number of active connections between the states i_1 and i_2 determines possible blocking (is there one connection less in either state). Hence, the difference in immediate costs, H_{i_1,i_2}, can be expressed as a function of i_1 and i_2.

Then, as the decision we choose the action $a_{i,k}$ leading to state j which minimizes the quantity

$$H_{i_0,j} + \hat{E}_{i_0,j} + \kappa \cdot \hat{\sigma}_{i_0,j}, \tag{4.15}$$

where κ is an adjustable parameter. Note that for $j = i_0$ this quantity is equal to 0. Thus, in order for another action j to replace the action i_0 of the standard policy, we must have $H_{i_0,j} + \hat{E}_{i_0,j} < -\kappa \cdot \hat{\sigma}_{i_0,j}$, i.e. we require a minimum level of confidence for the hypothesis $1_{j=i} \cdot \beta_k + v_j < 1_{i_0=i} \cdot \beta_k + v_{i_0}$. An appropriate value for κ has to be determined experimentally.

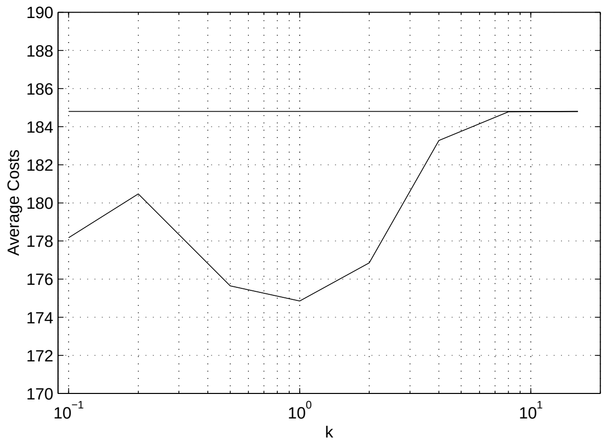

Figure 4.11: Average cost rate r of the iteration policy as a function of the parameter k. The horizontal line represents the cost rate of the standard policy. The minimum lies in the range $k = 0.5 \ldots 2$ in this case. The set of routes was specified with $\Delta l = 0$ and $rmax = 4$ explained in 5.2.1.

If κ is too small, wrong decisions are made more frequently. On the other hand too high a value of κ prevents the choice of other alternative actions totally. In Fig. 4.11 the performance behaviour of a certain system is depicted as a function of κ. The horizontal line represents the costs obtained with the standard policy. From the figure it can be seen that once κ is higher than about 10 the iteration policy reduces to the standard policy.

Time Complexity of the First Policy Iteration Approach

Clearly the simulation of the future, even for a limited period T, at each decision epoch makes the first policy iteration algorithm very time consuming. Assume that a single decision of the standard policy takes a constant time u. Let N be the number of the simulations that are run for each alternative action, A the average number of alternative actions per decision (possible RW pairs), λ the total arrival rate to the network (assuming uniform load for simplicity), and T the period covered by one simulation replication. Then, the running time of each decision is on average

$$u_i = A \cdot N \cdot (\lambda T) u = \lambda ANT \cdot u,$$

Algorithm 3 First policy iteration in D-RWA

A connection request arrives between nodes A and B
Let α_0 be the standard policy action, and $\alpha_1, \ldots, \alpha_m$ be alternative actions
Generate N independent sets of future arrivals, A_j
$D_0 \leftarrow 0$
for $j = 1$ to N **do**
    Make initial action α_0
    Run simulation with arrivals A_j
    Store the total costs to $C_0^{(j)}$ {Reference costs}
end for
for $i = 1$ to m **do**
    for $j = 1$ to N **do**
        Make initial action α_i
        Run simulation with arrivals A_j
        Store the total costs to $C_i^{(j)}$
    end for
    $\hat{E} \leftarrow \text{mean}(\mathbf{C}_i - \mathbf{C}_0)$
    $\hat{\sigma} \leftarrow \text{std}(\mathbf{C}_i - \mathbf{C}_0)$
    $H \leftarrow 1_{\alpha_i \text{ blocks}} \cdot \beta_k$
    $D \leftarrow H + \hat{E} + k \cdot \hat{\sigma}$
    if $D < D_0$ **then**
        $\alpha_0 \leftarrow \alpha_i$ {set new policy}
        $D_0 \leftarrow D$
    end if
end for
α_0 is the resulting policy

so the running time is λANT times longer than with the underlying algorithm. Neither λ nor A are parameters of the algorithm. Hence, the tradeoff between the goodness of solution and the running time is defined by choosing the value for product $N \cdot T$.

For example, to get decent results with a simple 11 node network (Fig. 5.1) with moderate load ($\mu = 1$ and $\lambda_k = 0.4$ for all κ), about 100 samples (simulation replications) were required, each $1/\mu$ time units long. So the increase in running time was of the order of $10^3 - 10^4$. It is clearly essential for the first policy iteration approach that the decisions of the underlying standard policy can be determined quickly.

4.6 Acceleration of Policy Iteration

In this section we study the possibility to accelerate the first policy iteration. In the first policy iteration N different future realisations are generated. The set of possible future events is huge (or actually non-countable) and only very small portion of them can be included in the simulations. Also the revenues or the costs of different realisations may differ a lot. Assuming we are estimating the cost due to the lost calls, we would like to favour such realisations which are more likely to cause costs. Essentially, it is not useful to sample such realisations which do not have any contribution to the costs. That is we are applying the importance sampling; we try to make the more important events more probable in the simulation.

Next a brief introduction to the importance sampling is presented, and then the technique is applied to the first iteration approach in order to reduce the number of required future replications (=samples) in the simulations[6].

4.6.1 Importance Sampling

Suppose we are trying to estimate the expectation of some random variable. In a problematic case the estimator based on direct simulations can have a high variance, which leads to many samples or a poor estimate. Such a problem can be avoided to some degree with appropriate variance reduction methods presented e.g. in [Ros00].

Let $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ be a set of random variables and $h(\mathbf{x})$ some function of them. We are interested in the mean of $A = h(\mathbf{X})$:

$$\theta = \mathrm{E}\left[A\right] = \mathrm{E}\left[h(\mathbf{X})\right].$$

[6]The importance sampling technique is also applicable when the performance of an arbitary heuristic algorithm is evaluated.

Assume that it is not possible to analytically calculate the above mean, but it is still possible to run simulations in order to get samples of A.

The obvious estimator for θ is obtained by directly taking k samples from the given probability distribution and averaging them,

$$\hat{A} = 1/k \cdot \sum_{i=1}^{k} A_i,$$

where A_i are i.i.d. random variables, $A_i \sim h(\mathbf{X})$. Then,

$$\begin{aligned} \mathrm{E}\left[\hat{A}\right] &= \theta, \\ \mathrm{V}\left[\hat{A}\right] &= \mathrm{E}\left[(\bar{A} - \theta)^2\right] = \mathrm{V}\left[A_i\right]/k. \end{aligned}$$

Thus, the smaller the variance of $\hat{A}$, the better. Several techniques to reduce the variance of the estimator are presented in [Ros00]. Here we concentrate on the importance sampling where the events having a greater contribution to the expectation are made more probable in the sampling and vice versa. This technique is supposed to give a better estimate with the same number of samples (or even fewer) than the direct estimator. Especially, those x, for which $h(\mathbf{x}) = 0$, can be excluded from the sample space as their contribution to final estimate is zero.

Let $f(\mathbf{x})$ be the probability density function of $\mathbf{X}$. Then,

$$\theta = \mathrm{E}\left[h(\mathbf{X})\right] = \int h(\mathbf{x}) f(\mathbf{x}) d\mathbf{x}.$$

The discrete case is treated identically, but instead of integration a n-fold summation is taken.

Let $g(\mathbf{x})$ be another probability density function for which it holds

$$g(\mathbf{x}) = 0 \qquad \Rightarrow \qquad f(\mathbf{x}) = 0,$$

i.e. whenever $g(\mathbf{x}) = 0$ the original probability density function $f(\mathbf{x})$ is also 0. Requirement quarantees that every possible event in the original distribution, i.e. an event that occurs with a non-zero probability, is also taken into account under the new probability distribution. Denote with $\mathbf{Y}$ a random variable whose p.d.f. is $g(\mathbf{x})$.

Then the quantity θ can be expressed as

$$\theta = \int \frac{h(\mathbf{x})\, f(\mathbf{x})}{g(\mathbf{x})} g(\mathbf{x})\, d\mathbf{x} = \mathrm{E}\left[\frac{h(\mathbf{Y})\, f(\mathbf{Y})}{g(\mathbf{Y})}\right] = \mathrm{E}\left[z(\mathbf{Y})\right], \tag{4.16}$$

where $z(\mathbf{x}) = h(\mathbf{x})\, f(\mathbf{x})/g(\mathbf{x})$. Thus, the interesting quantity θ can be estimated by generating samples of random variable $\mathbf{Y}$ with p.d.f. $g(\mathbf{x})$ and estimating the expected value of $z(\mathbf{Y})$ instead of $h(\mathbf{X})$.

Likelihood Ratio

The quantity $q(\mathbf{x}) := f(\mathbf{x})/g(\mathbf{x})$ in the equation (4.16) is called the *likelihood ratio*. When taking the samples of $\mathbf{Y}$ with the new p.d.f. $g(\mathbf{x})$ each outcome $h(\mathbf{x})$ is simply multiplied with the appropriate likelihood ratio $q(\mathbf{x})$ in order to get an unbiased estimator:

$$\theta = \mathrm{E}\left[h(\mathbf{X})\right] = \mathrm{E}\left[q(\mathbf{Y}) \cdot h(\mathbf{Y})\right]. \tag{4.17}$$

Furthermore, the average likelihood ratio $\bar{q}$ can also be determined:

$$\bar{q} = \mathrm{E}\left[q(\mathbf{Y})\right] = \mathrm{E}\left[f(\mathbf{Y})/g(\mathbf{Y})\right] = \int f(\mathbf{x})/g(\mathbf{x}) \cdot g(\mathbf{x})\, d\mathbf{x} = 1.$$

New Estimator with Importance Sampling

Instead of obtaining samples $A_i \sim h(\mathbf{X})$, we have another set of samples B_i, where $B_i \sim z(\mathbf{Y})$. The estimator of θ becomes

$$\hat{B} = 1/k \cdot \sum B_i,$$

for which it holds

$$\begin{aligned} \mathrm{E}\left[\hat{B}\right] &= \mathrm{E}\left[z(\mathbf{Y})\right] = \theta, \qquad \text{(i.e. the estimator is unbiased)} \\ \mathrm{V}\left[\hat{B}\right] &= \mathrm{V}\left[B_i\right]/k = \mathrm{V}\left[z(\mathbf{Y})\right]/k. \end{aligned}$$

The variance of the estimator is directly proportional to the variance of the estimated random variable.

Optimal Probability Density Distribution

If the new p.d.f. $g(\mathbf{x})$ can be chosen so that the variance of $z(\mathbf{Y})$ becomes smaller than the variance of the original estimator, then the importance sampling technique is worth using. The variances are

$$\begin{aligned} \mathrm{V}\left[h(\mathbf{X})\right] &= \mathrm{E}\left[h(\mathbf{X})^2\right] - \mathrm{E}\left[h(\mathbf{X})\right]^2 \\ &= \mathrm{E}\left[h(\mathbf{X})^2\right] - \theta^2, \\ \mathrm{V}\left[z(\mathbf{Y})\right] &= \mathrm{E}\left[z(\mathbf{Y})^2\right] - \theta^2. \end{aligned}$$

Thus, the optimal choice for new p.d.f. $g(\mathbf{x})$ minimizing the variance $\mathrm{V}\left[z(\mathbf{Y})\right]$ is the one which also minimizes the second moment of $z(\mathbf{Y})$.

4.6.2 Importance Sampling Applied to the First Iteration Approach

Next the importance sampling technique is applied to the first iteration approach while the aim is to get a reasonable estimate about the future with fewer samples leading to a faster decision making.

Assume that connection requests from each traffic class k, $k \in \mathcal{K}$, constitute a Poisson process with arrival rate λ_k. Let A_k be the random variable representing the number of class-k arrivals during the time interval $(0, T)$.

The Poisson process has the following nice property (see C.3):

Theorem 4.1 *Given that a certain number of arrivals from a Poisson process has occurred during a time interval* $(0, T)$*, these arrivals are uniformly and independently distributed in the same interval.*

This property will be useful when characterizing the importance sampling, or rather the likelihood ratio to be exact, within the first iteration framework.

The interesting quantity here is the average cumulative costs $\overline{c}$ during the finite time interval $(0, T)$:

$$\overline{c} = \mathrm{E}_f \left[c(\mathbf{X}) \right],$$

where $f(\mathbf{x})$ is the p.d.f. of the finite time future events (arrivals and departures) and $c(\mathbf{x})$ is the cumulative incurred costs during the time interval $(0, T)$ with the future realisation $\mathbf{x}$ of the process.

Altering Arrival Rates

The problem with simulating a typical network is that blocking is a rare event. In other words, if the offered traffic were higher the blocking probability would also be higher. So the obvious idea is to increase the arrival rates of some or all traffic classes, i.e. instead of using the original arrival rates $\{\lambda_k\}_{k \in \mathcal{K}}$ a new set of arrival rates $\{\lambda_k^*\}_{k \in \mathcal{K}}$ is used, where $\lambda_k^* > 0 \; \forall \, k \in \mathcal{K}$. The holding time distributions as well as the revenue rates / average losses of missed calls w_k are kept the same. Clearly the realisations generated with the new arrival parameters have new probabilities, or probability densities $g(\mathbf{x})$ to be exact.

Let $g(\mathbf{x})$ be the p.d.f. of the finite time future events with altered arrival rates. According to (4.17) the average cumulative costs during the time interval $(0, T)$ then become

$$\overline{c} = \mathrm{E}_g \left[q(\mathbf{X}) c(\mathbf{X}) \right],$$

where the $q(\mathbf{x})$ is the likelihood ratio. Thus, certain realisations with new arrival parameters are more likely to occur than they used to be, and vice versa.

Hence, the cost estimate obtained in a direct way with the new arrival process would give false results. To correct this we must weight the cost from each realisation appropriately with the likelihood ratio, which will depend only on the number of arrivals as will be seen in the following.

For the Poisson process determining the likelihood ratio is indeed fairly easy. The arrival realisations can be classified according to the number of arrivals from each traffic class. Let n_k be the number of class-k arrivals in a given realisation, i.e. $A_k = n_k$. For the Poisson process these arrivals were uniformly distributed to the given time interval $(0, T)$, as stated by Theorem 4.1. Thus, as every realisation $\mathbf{x}$ with the same number of arrivals from each traffic class k are equally likely to occur, we can concentrate on the number of arrivals and neglect the actual arrival times. A more formal proof follows.

Number of Arrivals

The probability that there are n_k class-k arrivals from the original arrival process A_k is

$$\mathrm{P}\{A_k = n_k\} = \frac{(\lambda_k \cdot T)^{n_k}}{n_k!} e^{-\lambda_k \cdot T},$$

i.e. a Poisson distribution with parameter $\lambda_k \cdot T$. The arrivals from the different traffic classes are independent and thus the probability of having $\mathbf{n} = (n_1, \ldots, n_K)$ arrivals from the original arrival processes is simply the product

$$\mathrm{P}\{\mathbf{A} = \mathbf{n}\} = \prod_{k \in \mathcal{K}} \frac{(\lambda_k \cdot T)^{n_k}}{n_k!} e^{-\lambda_k \cdot T} = T^n e^{-\lambda \cdot T} \prod_{k \in \mathcal{K}} \frac{\lambda_k^{n_k}}{n_k!},$$

where $n = \sum n_k$ and $\lambda = \sum \lambda_k$.

Similarly, the same number of arrivals from each traffic class with the new arrival process $\mathbf{A}^*$ would occur with the probability

$$\mathrm{P}\{\mathbf{A}^* = \mathbf{n}\} = T^n e^{-\lambda^* \cdot T} \prod_{k \in \mathcal{K}} \frac{(\lambda_k^*)^{n_k}}{n_k!}.$$

The costs we want to estimate can be written as

$$\overline{c} = \mathrm{E}\left[c(\mathbf{X})\right] = \mathrm{E}\left[\mathrm{E}[c(\mathbf{X})|\mathbf{A}]\right] = \mathrm{E}\left[\tilde{c}(\mathbf{A})\right],$$

where $\tilde{c}(\mathbf{n}) = \mathrm{E}[c(\mathbf{X})|\mathbf{A} = \mathbf{n}]$ is the average cumulative costs during the time interval $(0, T)$, when there are n_k uniformly distributed class-k arrivals, for each $k \in \mathcal{K}$, during the given period of time.

Similarly as in the continuous case, the importance sampling with arrivals $\mathbf{A}^*$ having a different point probability distribution becomes

$$\begin{aligned} \mathrm{E}\left[\tilde{c}(\mathbf{A})\right] &= \sum_{\mathbf{n}} \mathrm{P}\{\mathbf{A} = \mathbf{n}\} \cdot \tilde{c}(\mathbf{n}) = \sum_{\mathbf{n}} \frac{\mathrm{P}\{\mathbf{A} = \mathbf{n}\}\tilde{c}(\mathbf{n})}{\mathrm{P}\{\mathbf{A}^* = \mathbf{n}\}} \mathrm{P}\{\mathbf{A}^* = \mathbf{n}\} \\ &= \mathrm{E}_*\left[\frac{p(\mathbf{A}^*)}{p^*(\mathbf{A}^*)} \tilde{c}(\mathbf{A}^*)\right], \end{aligned}$$

where the subscript $*$ denotes that the expectation is to be taken with respect to the alternative point probability distribution of the arrivals $\mathbf{A}^*$. Similarly, the likelihood ratio $q(\mathbf{n})$ is

$$\tilde{q}(\mathbf{n}) = \frac{\mathrm{P}\{\mathbf{A} = \mathbf{n}\}}{\mathrm{P}\{\mathbf{A}^* = \mathbf{n}\}} = e^{-(\lambda_k - \lambda_k^*)T} \prod_{k \in \mathcal{K}} \left(\frac{\lambda_k}{\lambda_k^*}\right)^{n_k}.$$

Likelihood Ratio with Future Realisations

Next it will be shown that for any future realisation $\mathbf{x}$ the likelihood ratio $q(\mathbf{x})$ depends only on the number of arrivals from different traffic classes. To be exact, for each $\mathbf{n}$ it holds that

$$\forall\, \mathbf{x} \in \Omega_{\mathbf{n}} \qquad q(\mathbf{x}) = \tilde{q}(\mathbf{n}),$$

where $\Omega_{\mathbf{n}}$ denotes the class of future realisations with $\mathbf{n}$ arrivals from the respective traffic classes.

Formally, let $(\Omega, \mathcal{F}, \mu)$ be a probability triple, i.e. Ω is a sample space, $\mathcal{F}$ its σ-algebra and μ a probability measure $\mu : \mathcal{F} \to \mathbb{R}$ [Wil91, Dud89]. Here the sample space Ω consists of the possible arrivals and departures during the finite time interval $(0, T)$. The sample space is divided into sub spaces $\Omega_{\mathbf{n}}$ according to the number of arrivals n_k from each traffic class k, $k \in \mathcal{K}$. According to Theorem 4.1, the probability measure (or rather its density) μ is constant within each $\Omega_{\mathbf{n}}$.

There are two probability measures here: $\int f$ and $\int g$, which, like stated before, have a constant density within each $\Omega_{\mathbf{n}}$, i.e. also the likelihood ratio $q(\mathbf{x})$ is constant: $q(\mathbf{x}) = f(\mathbf{x})/g(\mathbf{x}) = c_f/c_g$ where c_f and c_g are some constants. Furthermore, it holds that

$$\begin{aligned} \mathrm{P}\{\mathbf{A} = \mathbf{n}\} &= \int_{\mathbf{x} \in \Omega_{\mathbf{n}}} f(\mathbf{x}) = \int_{\mathbf{x} \in \Omega_{\mathbf{n}}} c_f, \qquad \text{and} \\ \mathrm{P}\{\mathbf{A}^* = \mathbf{n}\} &= \int_{\mathbf{x} \in \Omega_{\mathbf{n}}} g(\mathbf{x}) = \int_{\mathbf{x} \in \Omega_{\mathbf{n}}} c_g. \end{aligned}$$

Thus, for each $\mathbf{x} \in \Omega_{\mathbf{n}}$ the likelihood ratio $q(\mathbf{x})$ becomes

$$q(\mathbf{x}) = \frac{c_f}{c_g} = \frac{\int_{\mathbf{x} \in \Omega_{\mathbf{n}}} c_f}{\int_{\mathbf{x} \in \Omega_{\mathbf{n}}} c_g} = \frac{\mathrm{P}\{\mathbf{A} = \mathbf{n}\}}{\mathrm{P}\{\mathbf{A}^* = \mathbf{n}\}} = \tilde{q}(\mathbf{n}).$$

Next the likelihood ratio is determined for some simple cases.

Constant Increase in Arrival Rates

One possibility to increase the expected costs is to increase all the arrivals rates by multiplying them with a common factor $\alpha > 1$

$$\lambda_k^* = \alpha \cdot \lambda_k.$$

Then the likelihood ratio $\tilde{q}$ is

$$\tilde{q} \;=\; \prod_{k\in\mathcal{K}} \left(\frac{\lambda_k}{\alpha\cdot\lambda_k}\right)^{n_k} e^{-(\lambda_k-\alpha\lambda_k)T} = C\cdot\left(\frac{1}{\alpha}\right)^n,$$

where $C = \prod_{k\in\mathcal{K}} e^{(\alpha-1)\lambda_k\cdot T} = e^{(\alpha-1)\overline{A}}$, $\overline{A} = \lambda T$ is the expected number of arrivals from the original process, and $n = \sum_k n_k$ is the total number of arrivals in a given realisation. So the likelihood ratio can be written as

$$\tilde{q} = \frac{\left(e^{\alpha-1}\right)^{\overline{A}}}{\alpha^n}.$$

The good thing with this choice is its simplicity, there is only one constant which must be determined.

Increase Proportional to the Revenue

Another possibility is to increase the arrival rates of traffic classes in proportion to their revenue:

$$\lambda_k^* = \frac{1}{\alpha} w_k\lambda_k,$$

where α is some positive constant to control the total arrival rate. Now the likelihood ratio $\tilde{q}$ is

$$\tilde{q} \;=\; \prod_{k\in\mathcal{K}} \left(\frac{\lambda_k}{\frac{1}{\alpha}\cdot w_k\cdot\lambda_k}\right)^{n_k} e^{(\lambda_k-\frac{1}{\alpha}w_k\lambda_k)T} = C\cdot\frac{\alpha^n}{w_1^{n_1}\cdot w_2^{n_2}\cdot\ldots\cdot w_K^{n_K}},$$

where n is again the total number of arrivals and constant C is

$$C = \prod_{k\in\mathcal{K}} e^{-(1-w_k/\alpha)\lambda_k T} = e^{-\overline{A}+\frac{T}{\alpha}\sum_{k\in\mathcal{K}} w_k\lambda_k} = e^{-\overline{A}+\frac{1}{\alpha}\overline{D}},$$

where $\overline{D}$ is the average offered revenue during the time period.

4.7 RWA with Additional Information

In this section we consider the case where some additional information is available. Among the possible extensions are

1. the durations of the existing and currently arriving connections are known
2. the durations of all connections are known (can be still class specific but constant)

3. the future is totally deterministic (the arrivals and their durations are known)

In the first two cases the arrival process is assumed to be a Poisson process. Also, any combination of traffic classes described above and ordinary Poissonian arrivals with exponential service times could be considered.

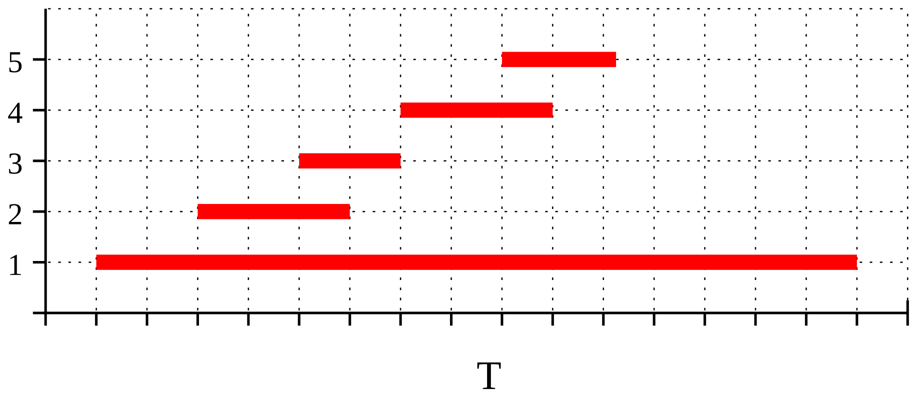

Figure 4.12: Sample arrival process to the two node network.

With the additional information the optimality may depend on the problem formulation, whether we are looking at the number of blocked calls or the average number of active connections over the time.

In the following a problem formulation with perfect information about the future is first presented. With perfect information the future is deterministic, and the problem is slightly different. Then the standard traffic model with Poissonian arrivals is considered, however assuming that the durations of connection requests are revealed at the time of arrival. A simple M/M/1/1-system is considered as an example of this type.

4.7.1 Perfect Information

As an example consider a simple two node network with one fibre connecting the nodes. In Fig. 4.12 a sample arrival process is depicted and it is assumed that everything is deterministic (perfect information about the future). Thus, the arrivals and durations of connection requests are known. The problem is one kind of *time tabbing* or *scheduling* problem, we try to pack the connections as efficiently as possible where time is an additional dimension. Assuming that there are two wavelengths ($W = 2$) available we have the following (essential) cases:

If the revenue is per call the optimal policy is clearly *accept 2,3,4,5*. On the other hand, if the revenue is collected per time unit the optimal policy is *accept 1,2,4*. Thus, it is strikingly clear that the optimal policy is different depending on if the costs/revenues are defined by calls or per time unit, unlike the case

policy	blocked calls	carried calls	carried traffic
accept 1,2,4	2	3	21
accept 1,2,5	2	3	20
accept 1,3,4	2	3	20
accept 1,3,5	2	3	19
accept 2,3,4,5	1	4	10

Table 4.2: The number of blocked calls vs. carried traffic.

with the standard traffic model. It is not worth accepting a long call with low payoff. That is, we are no longer working in the MDP framework as the policy α no longer defines a Markov system but a totally deterministic system.

When the first policy iteration approach is applied to the case of perfect information, we can go beyond the first iteration step, i.e. a decision tree can be built and after evaluating its leaves the most promising move is made (in infinite time horizon problem we cannot build the tree to infinity). The number of the future samples required is one as we *know* the future. Hence only one sample is necessary. Furthermore, if there are A alternatives on average at each decision epoch, the number of policies up to depth d is

$$C = A^d.$$

By depth we mean the number of levels in the decision tree, e.g. in the case of the first policy iteration $d = 1$. Assuming that there are only few choices (if more, consider only the most promising candidates using similar heuristic reasoning as standard policy) for example 4, a feasible depth d is of the order $8-10$. This corresponds to about $6 \cdot 10^4 \ldots 10^6$ different possibilities, which can be checked with simple brute force method.

Furthermore, while evaluating the previous move we have already obtained a subtree of the next step. Thus, only the lowest level of tree needs to be expanded one level further and then new estimates are calculated for each node.

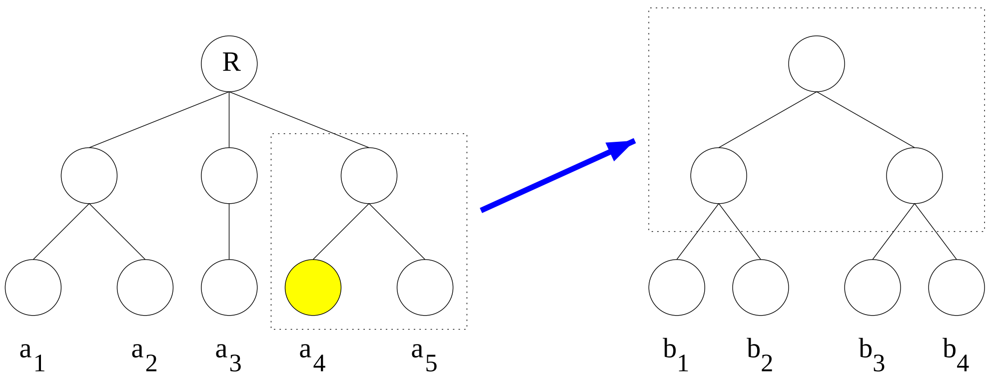

Figure 4.13: An example decision tree with the search depth $d = 2$. In the figure, values a_i denote the estimated revenues using given path, and a_4 is the maximum among them.

This is equivalent to building a decision tree and evaluating its leaves. In Fig. 4.13 such a tree is depicted. The root of the tree denotes the current state of the system and there are three possible choices to be made. A choice here is equivalent to deciding a route and assigning a wavelength to an incoming call (or rejecting it). In the example decision tree the search extends to two decisions, i.e. all possible combinations how the next two connection requests can be treated are considered. After that the future costs are estimated using the standard policy and, by combining these, an estimate for each path is obtained (values a_i). In combining it is probably useful to use discounting (giving less weight on later costs as the estimate becomes less accurate), because otherwise the last decision at the end of sample realisation has too much weight.

In the example presented in Fig. 4.13 the path a_4 turned out to be the best choice among all the paths a_i. So, the initial choice along this path is chosen as the next decision. Then a new tree corresponding to the path a_4 is formed, i.e. the new tree consists of the subtree of the previous tree with one additional level of possible choices so that the "search depth" remains constant 2. Then the same procedure is repeated again. Assuming that the standard policy is near optimal, the resulting policy can be assumed to be very good.

4.7.2 Known Durations

Here we consider a case where the arrival process of each traffic class is a Poisson process and the durations of connections are exponentially distributed, but there is some additional information available. Namely, at the time of arrival the duration is drawn from the exponential distribution and it becomes known to the controller. That is, at the time of its arrival the customer presents a request for a certain time interval. Thus, at any point of time the controller is aware of when the currently configured connections will end. Hence, the future is somewhat more deterministic than what it was with the standard traffic model.

If the policy does not take into account the known durations, the performance of the system is identical with a system with unknown durations, i.e. a normal MDP. In the first policy iteration the relative values of different actions were estimated by sampling the finite future with the standard policy. As we now know more about the future, this information can be exploited in the simulations. The knowledge about the durations of existing and current connections can be taken as an ability to sample "the more correct" future realisations. That is, we should get a better estimate of what indeed will happen. Furthermore, the first iteration policy with otherwise same parameters should give better results if the durations of connection requests become known at the moment of arrival.

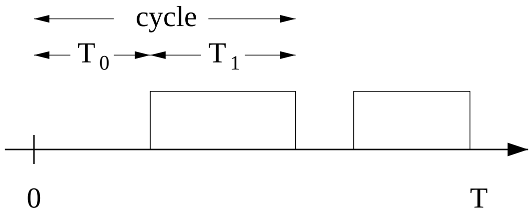

Figure 4.14: M/M/1/1-server model and one service cycle.

M/M/1/1-Server as an Example

Next a simple M/M/1/1-server model is studied in order to get some idea if the knowledge of the customers' service times actually could lead to considerable improvements. The simple model is chosen, since it is clearly extremely difficult to solve the optimal policy analytically for anything more complex.

Assume that we have a one server model without any waiting places. Customers arrive according to a Poisson process with intensity of λ, and their service times obey the exponential distribution with parameter μ. Furthermore, it is assumed that the service time becomes known to the system at the moment of arrival just before the acceptance / rejection decision.

The system can be in two states: either it is empty or there is a customer in service. If the policy is not to reject any customers, the blocking probability is

$$b_0 = \frac{\lambda}{\lambda+\mu} = \frac{a}{1+a}, \qquad \text{where } a = \lambda/\mu.$$

Consider an alternative policy that accepts only those customers whose service time is smaller than x. Hence, at the limit $x \to \infty$ the policy reduces to the previous *accept all*-policy. It is obvious that if a customer with service time x is rejected, then also any customer having a longer service time should be rejected, as accepting such a customer would cause even more blockings in the future on average.

Examine one service cycle that starts when a customer leaves the system and ends when the next (non-rejected) customer has left the system. Within this cycle, denote the idle time with T_0 and the service time with T_1 (see Fig. 4.14). The probability that a customer arriving at an empty system is accepted is

$$\mathrm{P}\{\text{customer accepted}|\text{empty system}\} = \mathrm{P}\{D < x\} = 1 - e^{-\mu x} =: p(x).$$

The number of customers arriving to an empty system during the idle time obeys the geometrical distribution[7] with parameter $p(x)$. Hence, the average number of blocked customers during the idle time is

$$\mathrm{E}[B_0] = \frac{1}{p(x)} - 1 = \frac{1-1+e^{-\mu x}}{1-e^{-\mu x}} = \frac{e^{-\mu x}}{1-e^{-\mu x}}.$$

[7] Service times are i.i.d., Bernoulli-trials until one succeeds.

The average service time of an accepted customer is

$$\mathrm{E}[D|D<x] \quad = \quad \frac{\int_0^x t\mu e^{-\mu t}\,dt}{\mathrm{P}\{D<x\}} = \frac{1}{\mu} - \frac{e^{-\mu x}}{1-e^{-\mu x}}x =: \mathrm{E}\left[T_1\right].$$

During the service time on average $\mathrm{E}\left[B_1\right] = \lambda \cdot \mathrm{E}\left[T_1\right]$ customers are rejected since the system is full. Hence, during one service cycle one customer is served and $\mathrm{E}\left[B_0\right]+\mathrm{E}\left[B_1\right]$ customers are blocked on average. Thus, the average blocking probability of the system is

$$\begin{aligned} b(x) &= \frac{\mathrm{E}\left[B_0\right]+\mathrm{E}\left[B_1\right]}{1+\mathrm{E}\left[B_0\right]+\mathrm{E}\left[B_1\right]} = \frac{\frac{\lambda}{\mu} + (1-\lambda x)\frac{e^{-\mu x}}{1-e^{-\mu x}}}{1+\frac{\lambda}{\mu} + (1-\lambda x)\frac{e^{-\mu x}}{1-e^{-\mu x}}} \\ &= \frac{a+z(x)}{1+a+z(x)}, \end{aligned}$$

where

$$z(x) = (1-\lambda x)\,\frac{e^{-\mu x}}{1-e^{-\mu x}} = \frac{1-\lambda x}{e^{\mu x}-1}.$$

The function $b(x)$ is strictly increasing. So in order to minimize the blocking probability, the function $z(x)$ must be minimized. That is, the optimal threshold x is the one which minimizes the function

$$z(x) = \frac{1-\lambda x}{e^{\mu x}-1}. \tag{4.18}$$

By using L'Hospital's rule we get

$$\begin{cases} \lim_{x\to 0^+} z(x) = \infty, \\ \lim_{x\to\infty} z(x) = 0. \end{cases}$$

Also $z(1/\lambda) = 0$. Furthermore, by examining the numerator and denominator it can be deduced that

when $x \in (0, 1/\lambda)$, then $z(x) > 0$,
when $x \in (1/\lambda, \infty]$, then $z(x) < 0$.

Thus, the optimal rejection threshold x is somewhere between $(1/\lambda, \infty)$, and can be numerically obtained by minimizing (4.18).

In Fig. 4.15 an example is presented. The offered load was $a = 1$ with $\lambda = \mu = 1$. The optimal threshold was found to be about 1.8414, which gives about 45.7% blocking probability. Hence, in this case the benefit from knowing the customer service times was about 9% reduction in the blocking probability. The original blocking probability was very high, i.e. 50%, which means that even if we block one customer the next one will arrive quite soon. Thus, the gain rejecting some of the customers is not very high in this case.

4.8 Summary

In this chapter the dynamic routing and wavelength assignment problem was studied. For any moderate size or larger network the state space is huge and

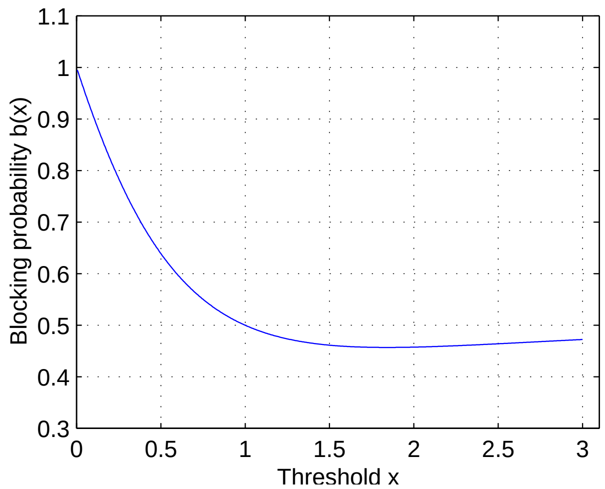

Figure 4.15: The blocking probability $b(x)$ with $\lambda = \mu = 1$ as a function of threshold x. The minimum blocking probability $b(x) \approx 0.457$ is obtained when $x \approx 1.84$.

the exact optimal policy cannot be obtained. Thus, the only practical solution is to use heuristic algorithms. Several reasonably good heuristic algorithms are described in the literature.

A novel way to improve any given heuristic is to apply so called first policy iteration step. The drawback with the first policy iteration is a considerably longer running time when compared to the standard policy. This problem can be alleviated to some degree by using the importance sampling method, where one tries to favour events having the largest contribution to the estimated quantity.

At the end of the chapter some special cases of the traffic process are considered, i.e. cases where some additional information is available about the traffic process (e.g. the duration of arriving connection request).

Chapter 5

Evaluation of Dynamic RWA Algorithms

The network operator's goal is generally to make as much profit as possible, which to some degree is equivalent to maximizing the average number of users in the network. On the other hand, the customers do not like if their requests get often blocked, which possibly can lead to a situation of losing some customers to the competitors and resulting in a decrease in connection arrival rates, which in turn reduces incomes.

Evaluation of different routing and wavelength assignment algorithms is not a straightforward task. For instance, some connection requests may have a higher priority than the others.

In this chapter simulation results are presented for different kind of traffic processes. Special attention is paid to the first iteration approach, presented in Section 4.5, and its performance in the different test cases. Throughout this chapter the traffic is assumed to be dynamic, i.e. traffic requests arrive according to some traffic pattern.

5.1 Blocking Probability vs. Offered Load

Suppose we are given a network and a traffic matrix, which defines the average number of lightpath requests per time unit between any pair of nodes. We have two algorithms A and B which are to be compared. Assume that traffic matrix can be scaled with a real coefficient $\alpha > 0$. We can define two gains (similarly as in [KA98]):

- *Blocking probability gain* defined as the difference between blocking probabilities of different algorithms with the same offered load

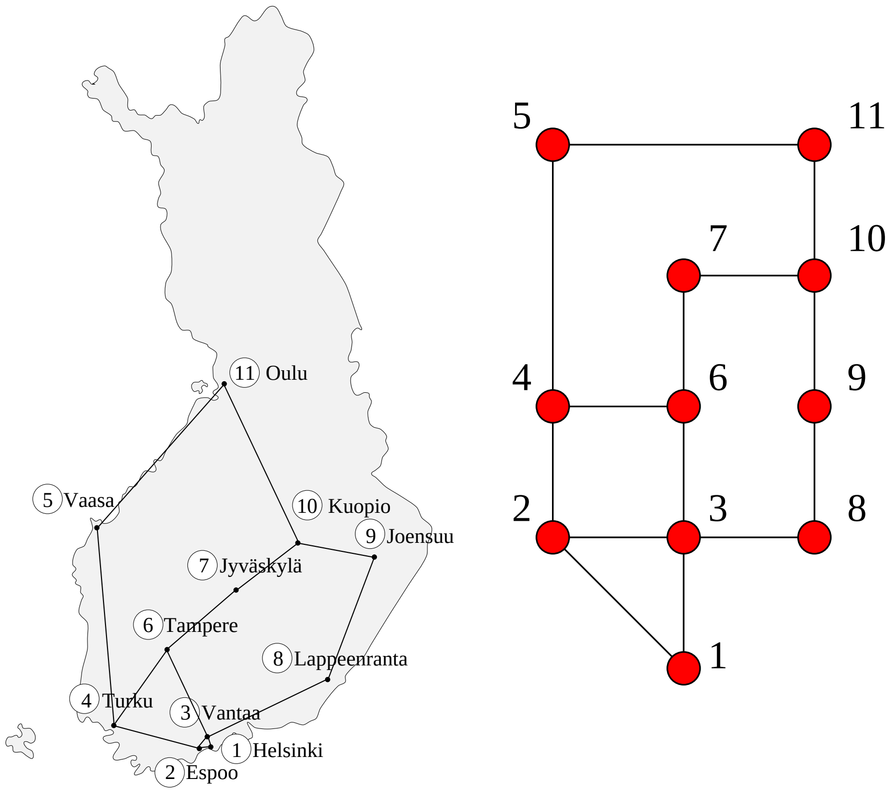

Figure 5.1: Hypothetical WDM-network residing in Finland.

- *Utilization gain* defined as the maximum increase in offered load possible while maintaining lower or equal blocking probability.

In the test cases presented in this chapter the performance of different algorithms is compared using the same offered load, i.e. the blocking probability gain is examined.

5.2 Simulation Parameters

The core network used in all simulations is presented in Fig. 5.1. The network is assumed to have 8 wavelength layers and no wavelength conversion is possible.

It should be recognized that the results for the first iteration policy were obtained by two levels of nested simulations. In order to assess the performance of the policy, an outer simulation is run, where connections arrive and leave the network and blocking times or events are recorded. Upon each arrival in this outer simulation, a number of inner simulations are launched from the current state in order to make a comparison between different decision alternatives. Based on this comparison one alternative is chosen and used in the outer simulation, which then continues until the next arrival upon which time the decision analysis by the inner simulations is again started.

5.2.1 Selection of Routes

The possible routes per node pair (or traffic class) were calculated beforehand. Generally the set of routes is enormous. So, some way of pruning is needed. In this study the set of routes was specified with parameters Δl and *rmax*. Parameter Δl sets the maximum number of extra additional links a route can contain when compared to the shortest route. The second parameter *rmax* defines the maximum number of routes per traffic class, i.e. only the *rmax* first found routes are included in the set. For example, with Δl=0 and *rmax*=10 only the shortest routes are included, and if there are more than 10 shortest routes for some node pair only the first 10 found are included.

A third possible parameter to limit the running time of the first iteration approach is *maxtest*, which defines the number of alternative actions evaluated against the standard policy. This effectively limits the running time when the load is low in the network and there are plenty of RW pairs available.[1]

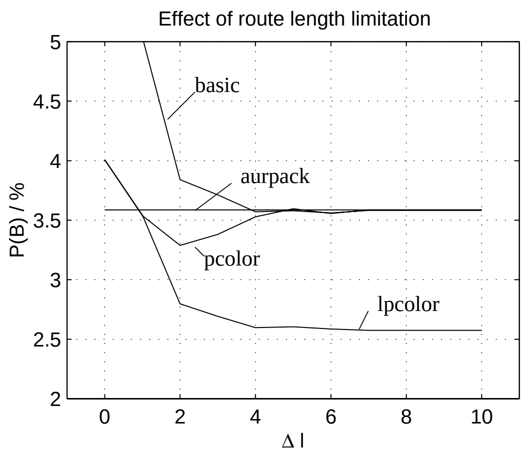

Figure 5.2: Blocking probability as a function of the parameter $\boldsymbol{\Delta l}$. The other routing parameter *rmax* set no limit on the number of routes. At $\boldsymbol{\Delta l = 3}$ the standard policies from the worst to the best are *basic*, *aurpack*, *pcolor* and *lpcolor* (all without the first iteration).

In Fig. 5.2 the performance of a few heuristic algorithms is presented as a function of the routing parameter Δl. The other routing parameter *rmax* was chosen to be high enough in order not to cause any restriction on the set of routes. The y-axis represents the blocking probability under a uniform load. Clearly too small a set of routes limits the performance but, as can be seen from the Fig. 5.2, also too large a set of routes can decrease the performance of some algorithms. In this case the problematic algorithm is *pcolor*, which needlessly favours the most used colours at the expense of longer routes, leading to degraded performance.

[1]Also when the load is low the choice of RW pair is not so critical

5.2.2 Estimation of an Optimal Simulation Period

Usually the longer the simulation period T is, the better results are obtained. Here we are, however, interested in how the current decision affects the results when the standard policy is used for all later decisions. As was explained before (see Fig. 4.10), after a transient period the cost rate $c_t(i)$ is very near to the long time average c of the standard policy. Simulating over a period longer than the duration of the transient thus gives no new information but actually only increases the noise resulting from the stochastic nature of the simulation.

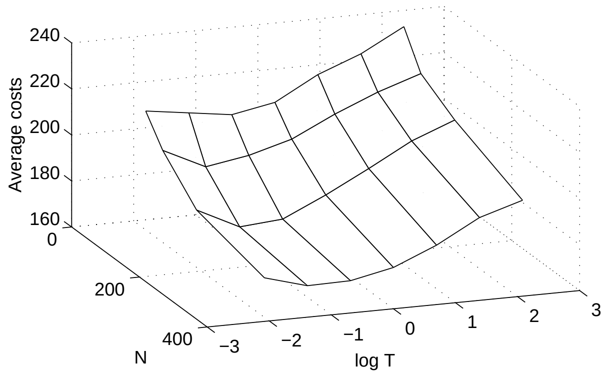

Figure 5.3: The performance of the first policy iteration algorithm with different simulations periods $\boldsymbol{T}$ and number of simulation replications $\boldsymbol{N}$. The traffic is uniform and the routing parameters are $\boldsymbol{\Delta l}$=0 and *rmax*=4. Algorithm *basic* is used as the standard policy. Load is $\boldsymbol{a = 0.4}$ for each traffic class. For reference, the standard policy alone gives an average cost of $\mathbf{260}$ for the same arrivals.

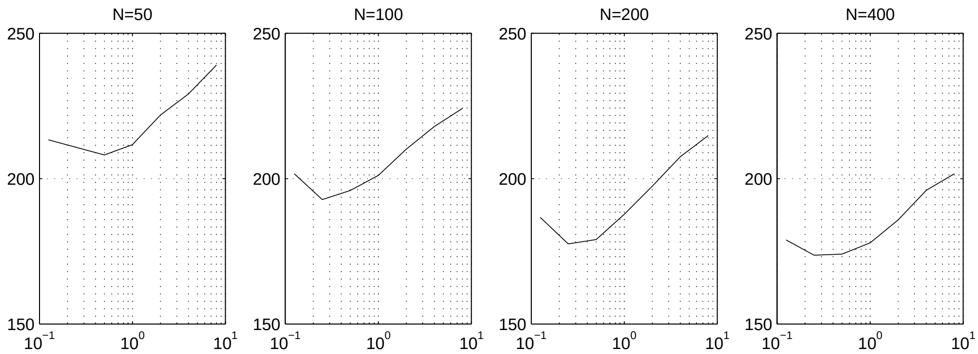

Figure 5.4: The performance of the first policy iteration algorithm as a function of period $\boldsymbol{T}$ for fixed number of simulation replications $\boldsymbol{N}$. The setup is the same as in Fig. 5.3; the graphs represent cuts of the 3D-surface of Fig. 5.3.

The average costs are depicted in Figs. 5.3 and 5.4 using the first policy iteration with different simulation periods and number of simulations. The *basic* D-RWA algorithm is used as the standard policy. The offered load to the network is $a = 0.4$ for each traffic class. In the mesh Fig. 5.3 the x-axis is $\log_{10}$ of the simulation period T and y-axis is the number of simulation replications N. The z-axis represents the average costs.

In Fig. 5.4 each subfigure has a fixed number of simulation replications ($N =$

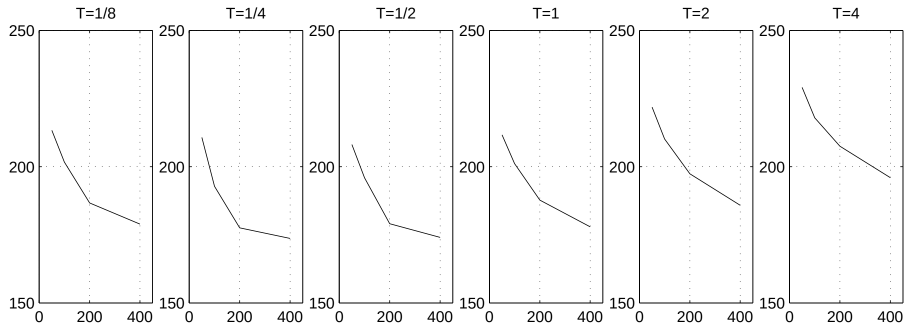

Figure 5.5: The performance of the first policy iteration algorithm as a function of the number of simulations replications $\boldsymbol{N}$ with fixed length simulation periods $\boldsymbol{T}$. The setup is the same as in Fig. 5.3 and 5.4 so the graphs represent cuts from the 3D-surface of Fig. 5.3

$50, 100, 200$ or 400). The x-axis represents the length of simulation period T and y-axis the average costs. Fig. 5.5, on the other hand, represents the performance of the algorithm with fixed simulation periods T as a function of simulation replications N. As can be seen from Figs. 5.3, 5.4 and 5.5 the results get worse as the simulation period T grows longer than $0.5 \ldots 1.0$ average holding times. This suggests that the optimal simulation period is about $0.5 \cdot 1/\mu$ in this case.

5.3 Symmetric Traffic and Costs

The set of simulations were run for the same test network that was used before, i.e. the network depicted in Fig. 5.1. The network was assumed to have 8 wavelengths on each link and the offered load was *uniform* among all node pairs. That is arrival rates λ_k, durations μ_k and weight factors w_k were the same for each traffic class k. Thus, knowing the average blocking probability of the system is enough in order to know the average cost rate of the system.

Suitable running parameters for the inner simulations for this system were estimated from Figs. 5.3, 5.4 and 5.5 (different representations of the same information). Based on this, the simulation period T was chosen to be $T = 0.25 \cdot 1/\mu$, and the number of simulation replications N was chosen to be $N = 50, \ldots, 200$ for each alternative action.

5.3.1 Numerical Results

Simulations were run with the quick heuristic algorithms as well as with the first policy iteration algorithm with different parameters and different standard policies. The resulting blocking probabilities are shown in Fig. 5.6. The upper part of the bars (light gray) represent two times the standard deviation and the mean value is in the middle of upper part. The routing parameters

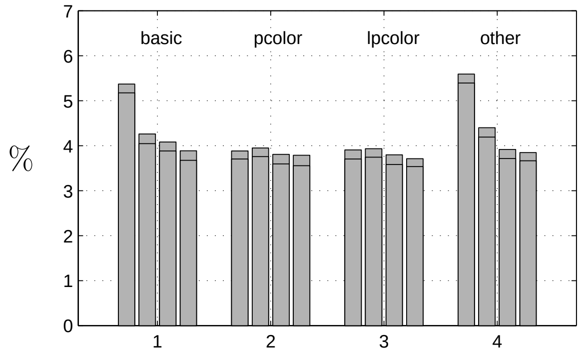

Figure 5.6: Blocking probabilities in percentages with quick heuristic algorithms and the first policy iteration. The routing parameters are: $\Delta l = 1$ and *rmax* $= 4$. Each of the three groups of bars, *basic*, *pcolor*, *lpcolor*, contain results obtained with standard policy and iteration policy with $N = 50, 100$ and 200. The last group of bars gives the results for *spread*, *porder*, *ll* and *aurpack*.

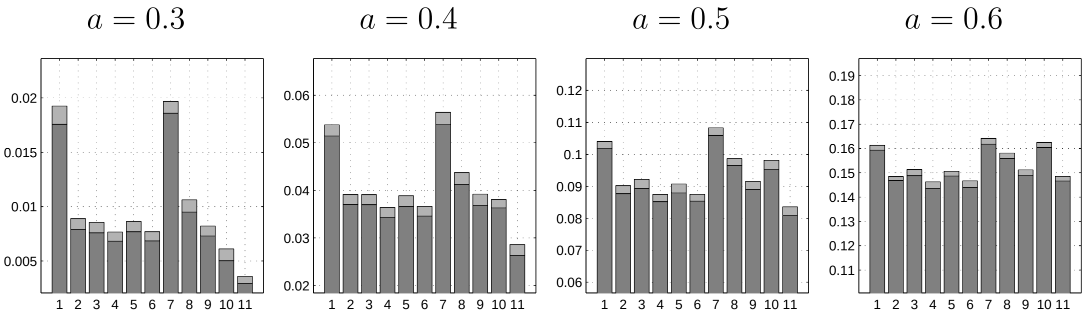

Figure 5.7: Blocking probability with loads ranging from $a = 0.3$ to $a = 0.6$. The set of routes were defined with Δl=1 and *rmax*=4. Algorithms are, from left to right, (1) *basic*, (2) *basic+iteration*, (3) *pcolor*, (4) *pcolor+iteration*, (5) *lpcolor*, (6) *lpcolor+iteration*, (7) *spread*, (8) *porder*, (9) *ll*, (10) *aurpack* and (11) *aurexhaustive*.

here were Δl=1 and *rmax*=4 which clearly limit the set of routes. The first group of bars represents the blocking probability with *basic* algorithm and the first policy iteration with $N = 50, 100, 200$ using *basic* as the standard policy. The second group is the same but using *pcolor* instead of *basic*, and similarly in the third figure the *lpcolor* is used. The fourth group is obtained with quick heuristics *spread*, *porder*, *ll* and *aurpack*.

The improvement obtained by the first policy iteration starting with the *basic* algorithm was notable, about 30%, while with *pcolor* the improvement is much less. Generally the results from iteration approach are always better than any of the heuristics which used the same set of possible routes. The *aurpack* and *aurexhaustive* use dynamic routing strategy without route length limitations and is here only for comparison. Also *ll*, *porder* and *spread* are presented just for comparison.

In another set of simulations the iteration approach was applied with different standard policies to get some idea about how important the underlying algorithm is. In the four diagrams of Fig. 5.7 the results can be seen with four different offered loads, with the blocking probability varying from quite a low to

Set	Description
I	*Uniform case.* The offered traffic between each node pair is uniform with the rate of $\lambda = 0.4$ and the cost of each lost request is equal to 1.0. The average duration of connection is $1/\mu = 1.0$.
II	*Non-uniform costs.* The offered load between each node pair is still uniform, but the cost of lost calls differs. A lost connection request to/from node 2 costs 3.0, connections 1-3, 1-6, 1-7, 6-7 and 6-10 have weight 1.0, and the rest of the connections have weight 0.5. So the offered income rate is the same as in the uniform case. The average duration of connections is the same 1.0.
III	*Non-uniform arrival rates.* Here we alter the arrival rates of the connections for which the other end is node 2 to 1.2, and the arrival rates between 1-3, 1-6, 1-7, 6-7 and 6-10 are set to 0.4, while the rest of the connections have an arrival rate of 0.2. Thus, the total arrival rate of the connection requests to the network remains again the same. The average duration of connections is the same 1.0 and cost of lost connection requests is 1.0 for all traffic classes.

Table 5.1: The test scenarios: case I is a reference point, where everything is uniform, whereas in the other two cases either arrival rates or costs are non-uniform.

a high value. The algorithms used were (in order) *basic*, *basic+iteration*, *pcolor*, *pcolor+iteration*, *lpcolor*, *lpcolor+iteration*, *spread*, *porder*, *ll*, *aurpack* and *aurexhaustive*. In these simulations the routing parameters were also the same Δl=1 and *rmax*=4. The number of simulations replications, N, for each alternative action was chosen to be 200. So *aurpack* and *aurexhaustive* have again much larger set of possible routes to choose from. It can be seen from the figure that in each case the iteration algorithm indeed gives slightly better results.

5.4 Asymmetric Traffic and Costs

In the previous section the traffic in the network was assumed to be homogeneous. Between each node pair there was a constant arrival intensity of connection requests and call durations obeyed the same probability distribution. Furthermore, each lost connection request contributed an equal cost. In practice the situation is rarely so simple. Therefore, in this section an asymmetric traffic process is studied and some numerical results from simulations are presented.

Three test scenarios were created, each having a different kind of characteristics. Every traffic scenario was based on the network shown in Fig. 5.1. The network was assumed to have 8 wavelengths available on each link and there was single fibre pair on every link. The test scenarios used in the simulations are listed in Table 5.1. The first scenario is uniform traffic, and it is used as a reference point. In the other two scenarios a special status is given to the node 2. The special status could arise e.g. in the case where the node represents a

parameter	value	description
W	22	the total offered income rate to the network, uniform in cases I and III
λ_{tot}	22	the total offered load to the network, uniform in cases I and II
μ	1.0	the mean duration of connection, $\sim$Exp(μ)
κ	2.0	constant related to the decision making, defines the level of "certainty"
N	200	the number of simulation replications in the iteration approach
T_1	$0.25 \cdot 1/\mu$	the length of simulation replication in the iteration approach, the first experiment
T_2	$0.50 \cdot 1/\mu$	the length of simulation replication in the iteration approach, the second experiment

Table 5.2: The running parameters used in test cases.

gateway to international network. To facilitate comparison with the uniform traffic case the rates λ_k and expected revenues per call w_k were adjusted so that the offered income rate $W = \sum_k \lambda_k w_k$ was kept constant.

5.4.1 Numerical results

In this section we investigate the performance of the first policy iteration starting from different heuristic policies. The parameters used in the simulations are given in Table 5.2. The choice of these parameters represents a tradeoff between the performance and the running time, and was done on the basis of the considerations of the previous sections.

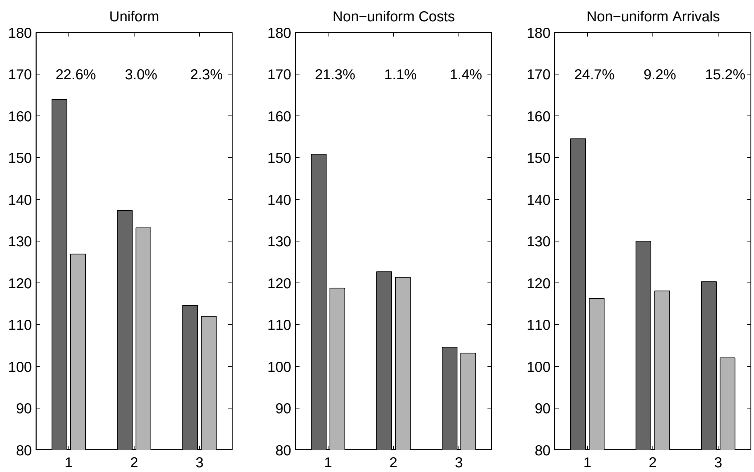

Figure 5.8: The performance of the algorithm in different test scenarios with (1) *basic*, (2) *pcolor* and (3) *lpcolor* as the standard policy. The simulation period is $\mathbf{0.25 \cdot 1/\mu}$. Each pair of bars relates to one of these policies, the left bar is obtained with the standard policy and the right bar with the corresponding first iteration policy.

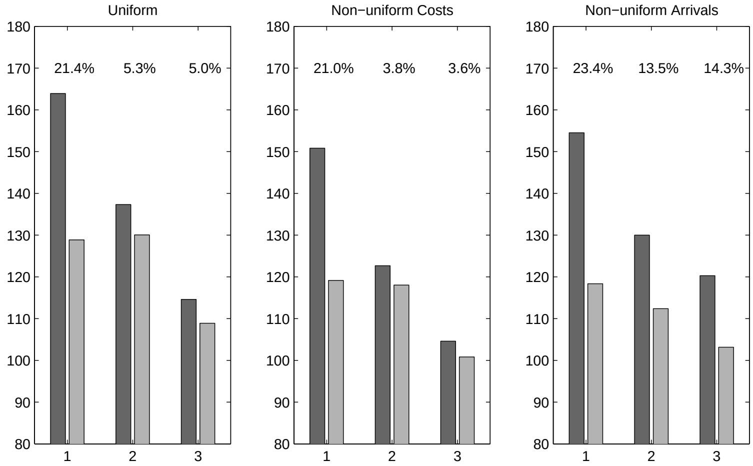

Figure 5.9: The performance of the algorithm in different test cases with (1) *basic*, (2) *pcolor* and (3) *lpcolor* as standard policy. The simulation period is $\mathbf{0.50 \cdot 1/\mu}$.

As suggested by Fig. 5.4 the number of simulation replications N was chosen to be 200 and the simulation period T to be $1/4$ or $1/2$ times the average holding time. The improvement when the number of samples was increased from 200 to 400 was not significant. Note that the previous simulations were run with a much smaller set of possible routes per node pair than these. The new routing parameters $\Delta l = 3$ and $rmax = 30$ allow routes that are longer than the shortest path in the search space. While the number of routes in these simulations is much larger than in those from which the parameters were obtained, it is expected that the same parameters work quite well. The value of κ was chosen to be 2.0 to be on the safe side (see Fig. 4.11).

The first policy iteration approach was studied with these parameters in three different traffic scenarios presented in the beginning of this section. The results with different parameters and algorithms are presented in Figs. 5.8 and 5.9. The y-axis represents the average costs obtained from the outer simulation. In Fig. 5.8 the simulation period T was $1/4$ times the average holding time, while in Fig. 5.9 the period was extended to $1/2$ times the average holding time. Each pane represents one traffic scenario. In the first pane the traffic is uniform, in the second pane the costs are non-uniform and in the last pane the traffic is non-uniform. The bars in figures represent average costs in each case. In each pair of bars the bar on the left represents the result obtained with the standard policy and the bar on the right is the one obtained with the first policy iteration. The standard policies from left to right are *basic*, *pcolor* and *lpcolor*. The results of the standard policies are naturally the same in both figures. The length of the outer simulation from which the average costs were collected was 200 holding times.

Figs. 5.8 and 5.9 show that in each case the iteration leads to a better policy, and that the improvement is really notable in the case where *basic* was used as the standard policy. Since *basic* is the worst algorithm, such an improvement is not a surprise. Note that in the uniform case the average costs are generally higher than in the other cases. This is probably because a large part of the “important” connections were quite short making the non-uniform case easier to control. It is also worth noting that the first iteration policy using either *basic* or *pcolor* as the standard policy gave similar results, while the best performance was obtained by using *lpcolor* as the standard policy.

In the non-uniform cost case the improvements were generally much less when *pcolor* or *lpcolor* were used as the standard policy. Still the difference between *pcolor* and *lpcolor* performance is clear and this suggests that the first policy iteration was not able to come to very close to the optimal policy.

The non-uniform arrival case on the other hand was very favourable for the first iteration approach. The improvement over any heuristic policy was around 10% or higher. This can be explained by the fact that as we are actually sampling the possible future realisations, the important (i.e. more probable) ones are automatically more frequently chosen. So we get a better grasp of the future with fewer samples. In the case where costs differ, similar favouring of more probable realisations does not occur naturally. It would probably be useful to study the applicability of importance sampling method to make more important paths more frequent in the simulations.

5.4.2 Robustness of the First Policy Iteration

The performance of the first policy iteration approach relies on good estimates of the parameters of the traffic process in contrast to the standard policies, which treat all the traffic classes equally. Poor parameter estimates may deteriorate the performance of the first policy iteration approach considerably and before using the first policy iteration the robustness of the algorithm should be studied carefully.

5.5 First Iteration with Additional Information

In this section we present simulation results of the case where additional information is available upon each arrival, namely the duration of the requested connection. The example network is the network depicted over Finland (see Fig. 5.1). The number of available wavelengths is 8 and the offered load is uniform $a = 1$ between each node pair. The cost function is the number of blocked calls, hence the length of the connection does not affect the price.

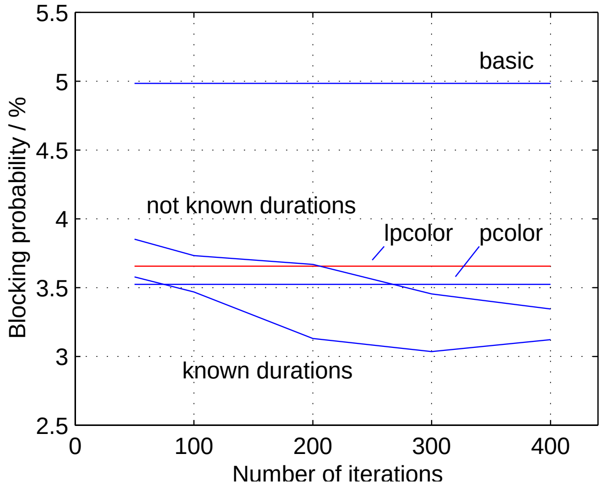

Figure 5.10: Simulation results with additional information. It can clearly be seen that the best performance is obtained with the first policy iteration approach with known durations. The x-axis is the number of simulation replications $\boldsymbol{N}$. The constant curves from top to bottom are *basic*, *lpcolor* and *pcolor*, and the other two curves are the first policy iteration using *lpcolor* as the standard policy with (lower curve) and without (upper curve) known durations.

5.5.1 Numerical Results

In Fig. 5.10 the blocking performance of different algorithms is presented. The horizontal lines are the standard policies *basic*, *lpcolor* and *pcolor* in that order from the worst to the best. The fact that *pcolor* outweighs *lpcolor* is probably due the route pruning parameters ($\Delta l = 1$ and $rmax = 4$). The upper non-constant curve is the first iteration approach using *lpcolor* as the standard policy without knowledge of the connection durations, and the lower curve is the first iteration using the same standard policy but with knowledge of the connection durations.

In the example, the improvement over the standard policy in the case of known connection durations is about 0.5% percentage units, i.e. the improvement in blocking probability is roughly about 16%. One can also notice that with the known connection durations fewer future replications N are required in order to achieve a similar performance as in the case where there is no information about the durations. Again, the first policy iteration approach has proven its robustness and ability to adapt to different situations.

5.6 Summary

In this chapter the dynamic RWA problem in all-optical networks has been studied, where the offered traffic, i.e. lightpath requests, followed different kinds of traffic processes. The problem was handled in the framework of

Markov decisions processes, except the last set of simulations, where connection durations became known at the moment of arrival. In particular, we studied the applicability of the first policy iteration approach, where the relative costs of states are estimated, as they are needed, by simulations on the fly. The problem with heuristic algorithms presented in the literature is that they do not take into account the non-uniform traffic or other peculiarities of the system. The first policy iteration is expected, to some extent, to overcome these deficiencies, and this was actually the case in all test scenarios.

The first case was *the uniform case*, where the offered traffic between each node pair had similar characteristics. Hence, the average blocking probability describes the overall performance of the system. The first iteration approach was capable to lower the average blocking probability up to 30% depending on the standard policy. Hence, the resulting policy was clearly better than the standard policies.

Then the traffic process was set to be a *non-uniform* or different traffic classes had *different revenue rates*. The heuristic algorithms do not take this kind of differences into account and it was hoped that the first policy iteration approach would overcome it, which indeed was the case. The performance improvement obtained by the policy iteration depends on the standard policy one starts with, and in these tests the average cost rate was reduced by about 10% to 20% in most cases. The improvement was not as high in the case where *pcolor* heuristics was used as the standard policy.

The last set of simulations were performed in order to see how the *additional information* affects the performance of the first policy iteration algorithm. In particular, even if the traffic process is totally deterministic the first iteration approach methodology is still applicable. Again the simple heuristic policies do not take advantage of additional information. The first policy iteration, on the other hand, uses the additional information naturally and the performance seen in the test case was superior to any of the simple strategies.

Thus, the robustness of the first iteration approach and its capability to adapt to different situation is very promising. On the other hand, the running time of the first iteration approach is probably too long for systems where decisions must be made in few seconds or even in milliseconds. But for slower systems it indeed can be used as an improvement to heuristic algorithms in real time. Even when a long running time makes the approach infeasible in real time, this method can still be used to assess, how close the performance of an arbitrary heuristic algorithm comes to the optimal policy.

Chapter 6

Restoration in WDM Networks

6.1 Introduction

In real world applications the protection against network failures plays an important role. Network operators want to make sure that the services they offer are reliable and protected against possible cable cuts or failures of different devices used in the network. The protection can be done in the optical domain. Then upper level protocols do not need to be aware of changes in the underlying network topology.

In this chapter the optical cross-connects (OXC) of the network are assumed to be reconfigurable, all the fibres are assumed to be bidirectional and the total traffic demand in the network is assumed to be constant. A very readable treatment of the restoration is given by S. Baroni in his Ph.D. thesis [Bar98].

One channel failures can occur if transmitter or receiver (of a channel) fails. Nonetheless, the result is a network where some links are not totally dead but can operate only to some degree. Then instead of reconfiguring all the connections using such links it is sufficient to reconfigure only those connections which are affected by the failure. As can be seen, the possible failures are numerous and will not be considered in more detail here.

6.2 Point-to-Point Link Protection

The simplest form of protection is point-to-point link protection, where each link is protected with one or more protection fibres. Traditionally, 1+1 protection, 1:1 protection, or 1:N protection are used [RS98, SB99]. In 1+1 protection the same traffic is transmitted at the same time using two fibres, while the destination end chooses the working fibre for reception. In all-optical networks

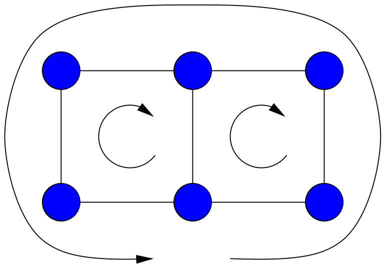

Figure 6.1: Construction of protection cycles.

this usually means 3 dB splitter loss at the transmitter end. The receiving end monitors the incoming signals and chooses the one which seems better.

The 1:1 protection scheme uses only one fibre to transmit the traffic at the time. Switching to the protection fibre is done after a fault is detected. If links are unidirectional only the receiving end detects a fibre cut, and thus there must be some protocol in use to acknowledge the transmitter about the cable cut. The 1:N protection scheme is a generalization of the 1:1 protection, where one protection fibre is shared between several working fibres simultaneously.

6.3 Protection Cycles

A quick local restoration algorithm is essential so that higher level protocols do not see any disruption in the service. Assuming that each link consists of four fibres, two for normal operation and two for protection, simple local restoration is possible using predetermined protection cycles.

Constructing the protection cycles for any planar graph is straightforward (see Fig. 6.1). The edges of each face[1] can be walked in clock-wise direction to form the cycles, and the cycle formed by the edges in the border of the graph can be walked in anti-clockwise direction. These cycles use each edge once for both direction. If some link is cut, there are always two independent protection cycles around it, which can be used to restore the cut lightpaths with very small delay.

Finding such protection cycles for an arbitrary graph can be harder. Note that orientable cycle double cover conjecture (see B.6) suggests that for any two-connected graph G such cycles exist.

[1]A graph face is the connected area which edges of the graph separate. The graph in Fig. 6.1 has two faces.

6.3.1 Planarity of Network

If the network is planar, i.e. it can be placed on a plane so that no fibres cross each other, then, as presented earlier, the construction of protection cycles is easy. Hence, it is worth checking if the given network is planar or not. Such an algorithm can be found in B.4.

6.4 Single-Fibre Network

The possible failures in an optical network include cable cuts and malfunctioning nodes. When a network cut occurs all connections using such a link must be re-routed. Depending on whether other connections are reconfigured as well or not we are led into two subproblems. On the other hand, when a node failure occurs all connections going through such a node must be reconfigured. Of course, connections to / from the malfunctioning node are impossible to fix. A node failure is roughly equal to the failure of all the links going into that node. In this thesis only link failures are considered, or more precisely, single link failures.

Optical networks can be divided into several categories according to the capabilities of the optical cross-connects, i.e. whether the nodes can perform a wavelength conversion or not. Also partial wavelength conversion is possible, but it is not considered here. Wavelength conversion makes routing and wavelength assignment easier but increases complexity of the nodes and thus makes them more expensive. If the network consists of wavelength interchanging cross-connects (WIXC), the re-routing problem is much easier as total number of users per link is the only limiting factor.

In the case of wavelength selective cross-connects (WSCX) the problem becomes harder. Here we assume wavelength agility, i.e. for each node pair active and restoration lightpaths can use different wavelengths.

The possible approaches in the case of a single link failure are

1. *Re-routing the whole network.* Such operation takes time and requires a centralized network management system (NMS). This is referred to as restore-all (RA) approach.

2. *Re-routing only failed connections.* This is a simpler operation than RA approach and does not affect other connections in the network. It is, however, likely to lead to greater number of wavelengths / fibres to guarantee protection of the services. This approach is referred to as the restore-only (RO) approach.

3. *Link restoration*. The failed link can be replaced with one or more possible reserve paths. This case was considered in Section 6.2.

Along with different cross-connects these lead to the following four cases:

1. WIXC-RA, wavelength interchange cross-connects and restore-all,
2. WIXC-RO, wavelength interchange cross-connects and restore-only,
3. WSXC-RA, wavelength selective cross-connects and restore-all,
4. WSXC-RO, wavelength selective cross-connects and restore-only.

The routing and wavelength assignment with restoration problem can be formulated at least in two ways. We can minimize the number of wavelengths while treating the number of fibres fixed, or we can minimize the number of fibres with fixed number of wavelengths W. In a single fibre case it is logical to limit the survey to the former case, i.e. the problem is to find out whether certain connections can be configured into the network so that any single link failure can be restored. If this is not possible then the next natural goal is to minimize the outage probability over all connections.

6.4.1 Lower-bounds for Wavelengths

Similarly as in the static routing problem presented in Chapter 3, some lower bounds for the number of required wavelengths are easy to obtain. One lower bound can be obtained by considering the average number of users using working links. After the removal of a single link j from the network, we can re-calculate the shortest paths for all the connections.[2] By dividing the sum of the path lengths by the total number of links left, $L-1$, we get the average number of users per link. This is clearly a lower bound to the number of wavelengths:

$$W \geq \max_j \frac{\sum_c l_{c,j}}{L-1},$$

where $l_{c,j}$ is the length of the shortest path for connection c when link j is not available.

Also the lower bound obtained in Section 3.3 for the static routing problem by cutting the network into two connected subsets can be modified to fit in the restoration case. Here, instead of dividing by the number of links crossing the cutting plane, the denominator is the number of links less one (one link is assumed to fail). Hence,

$$W \geq \max_{\text{cut}} \frac{\text{number of connections across cut}}{\text{number of links crossing the plane - number of non operational links}}.$$

It is worth noting that neither of these bounds depend on the used routing nor wavelength conversion capability. So they can be applied in every case.

[2] Note that only those connections which use malfunctional link are affected.

6.4.2 Relationship to the RWA without Restoration Problem

In this section the relationship between the static routing and wavelength assignment problem (S-RWA) presented in Chapter 3 and the restoration problem is considered. Assume that there is an algorithm which minimizes the number of wavelengths for a given network topology and static traffic demand [HV99]. This algorithm is referred to as the static routing and wavelength assignment algorithm and can be used to determine the number of wavelengths in the case where network restoration aspect is included. In every case, we must consider L different scenarios (any link can have a failure).

In the restore-all (RA) approach we can determine the number of wavelengths for each possible link failure and the maximum number of wavelengths required is the answer to question how many wavelengths are required to satisfy given traffic requirements with single fibre cut failures. Hence the original algorithm must be run L times and thus taking the restoration aspect into account only makes things more complex. The solution we get is as good as the used routing and wavelength assignment algorithm.

In the case of the RO approach we could first determine the best possible configuration for the network using the basic routing and wavelength assignment algorithm and then consider re-routing broken connections for each link at a time. This should be quite fast an operation compared to the original problem as we have only the maximum of W connections to re-route. But the configuration which could be optimal for S-RWA algorithm is not necessarily optimal when a link failure occurs, especially when WSXC nodes are used.

6.5 Multi-Fibre Case

In this case we have similar scenarios as in the single-fibre case. Here again we consider different cases of WIXC and WSXC nodes as well as different restoration strategies. The number of possible wavelengths W is given and the problem is to minimize the number of fibres (network planning). In the single-fibre case we optimized the number of wavelengths as it made no sense to optimize the number of fibres.

It is assumed that only those connections affected by the link failure are reconfigured, i.e. the RO approach. Baroni [Bar98] lists the following restoration cases in his Ph.D. thesis:

1. *Edge-disjoint path restoration with reserved capacity*. Each connection is assigned an active lightpath and a fixed edge-disjoint restoration lightpath. Thus 50% of the network capacity is unused under normal circumstances.

2. *Edge-disjoint path restoration*. Similar to the previous case but the restoration capacity is shared among other restoration paths.

3. *Path restoration*. The most flexible case, where any possible path is assigned to broken connections, i.e. any active and restoration paths may share links and capacity.

4. *Link restoration*. With WIXC nodes this approach is the only local approach as interrupted connections are re-routed around the failed link. So each used wavelength in each link has some predetermined restoration path which goes around the link. In the case of WSXC the wavelength translation is not possible and the restoration of the inoperative connections requires co-operation of both ends, i.e. basically a new connection is established.

Similar lower bounds as presented in Section 6.4.1 can be used in the multi-fibre case as well. Here we must, however, take into account that the number of fibres on links varies.

6.6 Network Planning with Restoration

In Section 3.4 the network planning problem was formulated in the static traffic case. When the restoration demands are taken into account the number of required fibres can only grow. Here the network is assumed to consist of WSXC nodes. The equation (3.5) must be modified so that each possible network cut, i.e. link failure, is taken into account. The model for network failure considered here assumes that one *link* (denoted with ℓ^*) at a time becomes inoperative and the rest of the network remains fully operational. If there are multiple fibres on some link, the model assumes that all of them become inoperative at the same time.

The connections are indexed with $z \in \mathcal{Z}$. Configuration of connections z into the network is equal to assigning the connection a route wavelength pair $(\,p(z),\, c(z)\,)$. Possible different configurations due to the link failures are denoted with subscripts.

Denote with $\mathcal{B}_{\ell^*}$ the set of connections which must be rerouted if link ℓ^* becomes inoperative. Formally,

$$\mathcal{B}_{\ell^*} = \{z \mid \ell^* \in p_z\}.$$

Clearly for all the links ℓ^*

$$\|\mathcal{B}_{\ell^*}\| \leq m_{\ell^*} \cdot W,$$

where m_ℓ is the number of fibres on link ℓ. In Section 6.5 four different cases were listed and will be briefly presented here again. In every case, the number of installed fibres define the cost of the solution and equation (3.4) can be used to obtain it.

The restore-all approach is not considered here as it is simply a matter of configuration of a network where one link is removed.

Edge-Disjoint Path Restoration with Reserved Capacity

In this case for each connection $z \in \mathcal{Z}$ primary and protection routes are needed. Formally, the solution is a pair of mappings $\{f_1, f_2\}$, where

$$f_1, f_2 : \mathcal{Z} \to \mathcal{A}_z \times \mathcal{W},$$

with additional requirement that the routes are edge-disjoint:

$$\forall z : p_1(z) \cap p_2(z) = \emptyset. \tag{6.1}$$

The constraint guarantees that whenever a protection route is needed, it is also fully functional. Furthermore, we get that the number of fibres needed is

$$m_\ell = \max_{c \in \mathcal{W}} \left(\sum_{\substack{z \in \mathcal{Z} \\ c_1(z)=c}} I(\ell \in p_1(z)) \;+\; \sum_{\substack{z \in \mathcal{Z} \\ c_2(z)=c}} I(\ell \in p_2(z)) \right),$$

where the first and second sums correspond to primary and restoration channels from the network, respectively. The total cost can be obtained by formula (3.4).

Edge-Disjoint Path Restoration

Now the restoration capacity is shared with normally edge-disjoint connections. Formula (6.1) must still hold, but the total number of fibres needed is now

$$m_\ell = \max_{\ell^* \in \mathcal{L}} \max_{c \in \mathcal{W}} \left(\sum_{\substack{z \notin \mathcal{B}_{\ell^*} \\ c_1(z)=c}} I(\ell \in p_1(z)) \;+\; \sum_{\substack{z \in \mathcal{B}_{\ell^*} \\ c_2(z)=c}} I(\ell \in p_2(z)) \right).$$

It can be expected that this case gives better results than the previous case.

Path Restoration

In this scheme all the available capacity in the network is used to reroute disconnected lightpaths. Instead of one restoration path, each connection can use different paths and wavelengths depending on which link has failure. Thus, the solution consists of $L+1$ (or L) different configurations:

$$f, f_1, \ldots f_L : \mathcal{Z} \to \mathcal{A}_z \times \mathcal{W},$$

with the constraint

$$\forall z, \ell^* \ : \ \ell^* \notin p_{\ell^*}(z),$$

i.e. the alternative routes must not use the inoperative link.

Thus the number of required fibres becomes

$$m_\ell = \max_{\ell^* \in \mathcal{L}} \max_{c \in \mathcal{W}} \sum_{\substack{z \in \mathcal{Z} \\ c_{\ell^*}(z) = c}} I(\ell \in p_{\ell^*}(z)), \tag{6.2}$$

with an additional constraint,

$$\forall z, \ell^* \ : \ \ell^* \notin p(z) \ \Rightarrow \ f(z) = f_{\ell^*}(z).$$

The introduced constraint guarantees that only inoperative lightpaths are reconfigured.

Another way to write (6.2) is the following:

$$m_\ell = \max_{\ell^* \in \mathcal{L}} \max_{c \in \mathcal{W}} \left(\sum_{\substack{z \notin \mathcal{B}_{\ell^*} \\ c(z) = c}} I(\ell \in p(z)) \ + \sum_{\substack{z \in \mathcal{B}_{\ell^*} \\ c_{\ell^*}(z) = c}} I(\ell \in p_{\ell^*}(z)) \right), \tag{6.3}$$

where the summation is done in two parts, first operative connections and then inoperative.

Link Restoration

By link restoration we mean here a concept where primary and restoration paths are identical except that in the restoration path the inoperative link is replaced with a (usually short) route around the failed link. Different wavelengths can use different around routes in order to achieve higher efficiency. The original lightpath as well as the restoration lightpath must, however, use the same wavelength in order to achieve a localised solution[3]. So in this case

[3]By localised solution we mean that only the nodes at the ends of the inoperative link must be reconfigured, which is not the case if wavelength changes.

only the nodes involved with the link restoration need to be reconfigured and the rest of the network need not be aware about the failure.

Again the solution is a set of $L + 1$ mappings

$$f, f_1, \dots f_L : \mathcal{Z} \to \mathcal{A}_z \times \mathcal{W},$$

with constraints

$$\begin{aligned} \forall z, \ell^* \quad &: \quad \ell^* \in p(z) \;\Rightarrow\; p(z) \cap p_{\ell^*}(z) = p(z) \setminus \{\ell^*\}, \\ \forall z, \ell^* \quad &: \quad c(z) = c_{\ell^*}(z). \end{aligned}$$

The constraints guarantee that the solution is local, and that the restoration paths do not use the failed link.

6.7 Summary

The restoration aspect increases the complexity of optimizing the network configuration, but it is essential in order to provide reliable connections. Restoration formulations can be divided into two cases depending on whether the rerouting is a viable option or not. Furthermore, the restoration can be accomplished with one protection path or with many paths. In this chapter the formulations assumed that only the inoperative connections were reconfigured.

Chapter 7

Conclusions

All-optical wavelength routed networks play probably an important role in the future data networks. The huge capacity they offer together with the high scalability makes them a very attractive choice for the next generation networks. Linear Lightwave Networks is another approach where the routing nodes combine incoming signals linearly, as the name suggests. In such a network forming a broadcast tree is quite natural. Another important technique to exploit the huge capacity of the optical fibre is the broadcast and select network, where time division multiplexing together with some MAC protocol can be used to offer good performance figures. Generally the broadcast and select networks are suitable for LAN applications. In this thesis, however, we have considered only the wavelength routed networks. Nonetheless, the promising future of WDM networks makes them an interesting topic for the research.

In Chapters 1 and 2 the WDM-technology together with current solutions was briefly described as a background material. The technology sets constraints such as the number of wavelengths to the optimization problems presented in this thesis. Then in Chapter 3 the static routing and wavelength assignment problem was shortly discussed. The chapter contains two alternative problem formulations and describes how those problems can be approached. All the first three chapters serve as an introduction to the main topic of the thesis, namely the RWA problem with dynamic traffic in wavelength routed networks.

In practice, the connections in the network change dynamically leading to a routing and wavelength assignment problem under dynamic traffic process. The optimal policy cannot be obtained due the enormous size of the state space. Still, several fairly well working heuristic algorithms are known. The main contribution of this thesis is the application of the first policy iteration to the dynamic routing and wavelength problem. It is known that the first policy iteration often comes very near to the optimum. This seems to be the case also here. In all the test cases the first iteration policy obtained a better policy than the so called standard policy, which can be chosen freely and serves as a starting point for the policy iteration step. The drawback with the first policy iteration approach is the huge increase in running time as the possible actions are evaluated by a set of on-line simulations. Whether this is a problem or not in reality depends on the time scale. Simulation acceleration techniques such as importance sampling can be used to reduce the running time of the algorithm. The dynamic RWA problem is tackled in Chapter 4 and some simulation results were presented in Chapter 5.

In the last chapter the protection and restoration aspects were briefly described. Restoration is an important issue especially in WDM networks, where even a short outage in service means interruption of a large number of connections because of the very high capacity of WDM links. Thus quick restoration schemes are required in order to be able to offer a good quality of service to the customers.

An interesting subject for the future work could be a further experimentation with the importance sampling with the first policy iteration approach and other means to reduce the running time of the algorithm. The choice of the importance sampling parameters determines how well the approach can work. Also robustness of the algorithm deserves further study. Another interesting (and endless) branch of possible studies is the RWA problem with various a priori information models about the traffic process such as known durations, different distribution for the duration etc. and any mixture of them.

Appendix A

Notations

A.1 The O-notation

When algorithms are compared it is feasible to use the "big-oh"–notation (see e.g. [Wei97, Knu68]):

Definition A.1 *A function* $f(n)$ *is* $O(g(n))$ *if there are positive constants* M *and* n_0*, such that*

$$|f(n)| \leq M|g(n)|, \; \forall n \geq n_0.$$

In particular, polynomial algorithms are characterized by the fact that their complexity (or steps run) is of order $O(x^m)$, where m is some positive integer.

A.2 NP-completeness

A problem is said to belong to the class of P problems if there exists a deterministic algorithm which solves it in polynomial time. Problems belonging this group can generally be considered as easy problems.

A problem is said to belong to the class of NP problems if there exists a non-deterministic machine which can solve the problem by a "brilliant guess", i.e. there exists an algorithm which can check the validity of given solution to the problem within polynomial time. Problems belonging to class P are clearly a subset of class NP. Generally all decidable problems belong to class NP.

The subset of NP-complete problems of NP can be considered as hard. A problem belongs to NP-complete problems if any problem in NP can be polynomially reduced to it. It is unknown if NP-complete problems have a polynomial time solution or not. Examples of an NP-complete problem are the graph node colouring problem (see B.5), finding a maximal clique of an arbitary graph, and the travelling salesman problem (TSP). In TSP one is asked to find the minimal cost cycle that goes through all the nodes of the graph (see e.g. [Wei97]).

Appendix B

Graph theory

A graph is a mathematical construction of vertices (or nodes) $v_i, v_i \in V$, which are connected by edges $e_i, e_i \in E$. Formally,

$$G = \{V, E\} .$$

The edges can be directional or bidirectional (which is perhaps a more usual case). Depending on the type of the edges, the graph is called either a (undirected) graph or a directed graph (or digraph) . In case of directed graph the edges are called arcs as they have direction.

A graph is a *weighted* graph if the nodes or the edges have a scalar attribute called the weight. This could represent e.g. the distance between network nodes.

B.1 Definitions

The *degree* of a node v_i, denoted by $d_i = \delta(v_i)$, is the number of edges connected to the node. The degree of graph G is the the maximum degree of its nodes: $\Delta(G) = \max_i d_i$.

A graph G is *complete* if there is an edge between all node pairs. In Fig. B.1 a complete graph containing five vertices is depicted.

A *complement* of graph G is the graph where there is an edge only between those nodes which are not neighbours in the original graph. The complement graph is denoted by $\bar{G}$. In particular, a graph sum of a graph and its complement is a complete graph.

A subgraph of graph $G = (V, E)$ consists of a subset of nodes $V' \subset V$, and such edges $e_i \in E$ where both ends belong to V'.

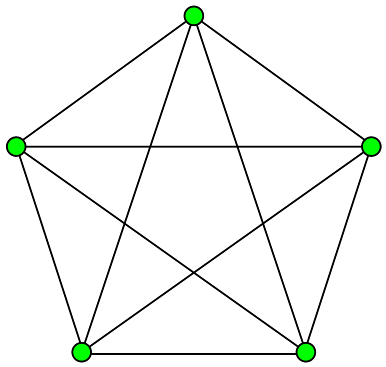

Figure B.1: K_5

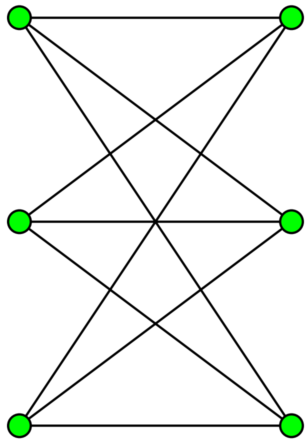

Figure B.2: $K_{3,3}$

A *path* (or walk) is a ordered list of nodes where subsequent nodes are neighbours. A path is a *cycle* (or circuit) if the first and the last node of the path are the same.

An edge e is called a *bridge* if there is no cycle where e belongs.

B.2 Shortest Path Algorithms

If the edges of a graph have weights, then the length of the path is the sum of the weights. Otherwise, each edge along the path is counted as length one.

There are two well-known algorithms to find the shortest path between node pairs, namely Dijkstra's algorithm and the Floyd-Marshall algorithm (see e.g. [BG92, Wei97]). Dijkstra's algorithm finds the shortest path between a given pair of nodes while the Floyd-Marshall algorithm finds the shortest paths between all node pairs. Both algorithms have the same complexity $\mathcal{O}(|V|^3)$ when used to find all node pairs, but the constant factor in the Floyd-Marshall algorithm is lower in dense graphs. Dijkstra's algorithm is presented in Algorithm 4.

B.3 Cliques and Independent Sets

A *clique* is a complete subgraph, i.e. a subset of nodes which all are neighbours to each other. An *independent set* is a subset of nodes which are not neighbours. Clearly an independent set of an arbitrary graph is a clique of the graph's complement graph, and vice versa. A maximal clique and maximal independent set are the ones that have the largest number of nodes in it. Finding a maximal clique or independent set from an arbitrary graph is an

Algorithm 4 Dijkstra's shortest path algorithm for weighted graphs

```
D̄ ← {∞, ∞, ... ∞, 0, ∞, ...} {distances from node A}
K̄ ← {0, 0, ... 0, 1, 0, 0, ...} {other distances are not known}
P̄ ← {0, 0, ... 0, 0, 0, 0, ...} {path is not know}
loop
  v ← arg min_i D_i {vertex with smallest known distance}
  if v is target vertex B then
    break
  end if
  K_v ← 1 {distance to v is now known}
  for w is adjacent to v do
    if K_w = 0 then
      if D_v + d(v, w) < D_w then
        D_w ← D_v + d(d, v) {new shortest path}
        P_w ← v {record path too}
      end if
    end if
  end for
end loop{then print out path backwards}
v ← A
while v ≠ B do
  print v
  v ← P_v
end while
```

NP-complete problem [KS99]. Still there are some algorithms for generating all the cliques/independent sets of an arbitrary graph (see e.g. [Öst99])

B.4 Planarity of Graphs

Definition B.1 *A graph G is* planar *if its vertices and edges can be drawn in a plane so that no two edges cross anywhere. A planar embedding of graph G is a mapping of the graph into the plane so that the edges do not cross.*

One necessary and sufficient condition for planarity of a graph is by Kuratowski in 1930 [SK77]:

Theorem B.1 (Kuratowski) *A graph G is non-planar if and only if G contains a subgraph which is homeomorphic to either K_5 or $K_{3,3}$.*

Vertices v_1 and v_2 belong to the same *bridge* of H in G (H is a subgraph of G) if there is a walk W between nodes so that each intermediate vertex of walk W does not belong to H. For a given subgraph H of G there can be several bridges. Especially in the case of planar graphs, if H is a cycle bridges are either inner or outer bridges. An attachment of bridge B is the set of vertices belonging to both B and H.

Let $\tilde{H}$ be a planar embedding of H and B some bridge. By $F(B, \tilde{H})$ we mean the set of faces where a bridge can be drawn without violation the planarity.

Algorithm 5, by Demoucron, Malgrange and Pertuiset from 1964, can be used to check planarity of given graph. The proof that the presented algorithm indeed works can be found e.g. in [BM76].

B.5 Node Colouring

The node colouring is a process where each node is given a colour (a labelled graph) so that no adjacent nodes have the same colour. A graph is said to be k-colourable if there exists a colouring with at most k colours. Generally, the node colouring problem is to find a legal colouring with minimum number of colours. Also this problem is known to be NP-complete. The minimum number of colours for which a legal colouring exists is called the *chromatic number* of graph. For example K_5 is clearly 5-colourable while $K_{3,3}$ is 2-colourable.

There is a famous theorem for colouring the planar graphs:

Theorem B.2 (Four-colour theorem) *Every planar graph is four-colourable.*

Algorithm 5 Planarity check of graph (Demoucron, Malgrange and Pertuiset, 1964)

let G_1 be a cycle in G
find a planar embedding of $\tilde{G}_1$ of G_1
$i \leftarrow 1$
while $\mathrm{E}(G) \backslash \mathrm{E}(G_i) \neq \emptyset$ **do**
  determine all bridges B_j on G_i in G
  for all B_j **do**
    find the set $F(B_j, \tilde{G}_i)$
  end for
  if there is a bridge B for which $F(B, \tilde{G}_i) = \emptyset$ **then**
    stop as G is non-planar
  else if there exists bridge B such that $|F(B, \tilde{G}_i)| = 1$ **then**
    let $\{f\} = F(B, \tilde{G}_i)$
  else
    let B be any bridge and f any face such that $f \in F(B, \tilde{G}_i)$
  end if
  choose a path $P - i \subseteq B$ connection two vertices of attachment of B to G_i
  $G_{i+1} \leftarrow G_i \cup P_i$
  obtain a planar embedding $\tilde{G}_{i+1}$ by drawing the P_i in the face f of $\tilde{G}_i$
  $i \leftarrow i + 1$
end while

The four colour theorem states that any planar map is four-colourable. By colouring a planar map we mean assigning each area a colour such that no areas sharing non-zero length border have same colour. Clearly colouring a planar map is equivalent to colouring a planar graph The theorem was first proven by Kenneth Appel and Wolfgang Haken in 1976. The proof of the theorem is long and was done with the aid of long computers runs. A comprehensive introduction to the theorem and its interesting history can be found from [SK77, AH77].

Over the years several algorithms have been proposed to find a suboptimal colouring for an arbitrary graph [Mit76, Bré79, HdW87]. A simple greedy algorithm is described in Algorithm 6. The algorithm simply gives each node

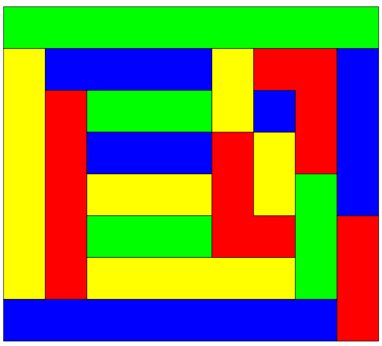

Figure B.3: A sample planar map coloured with four colours.

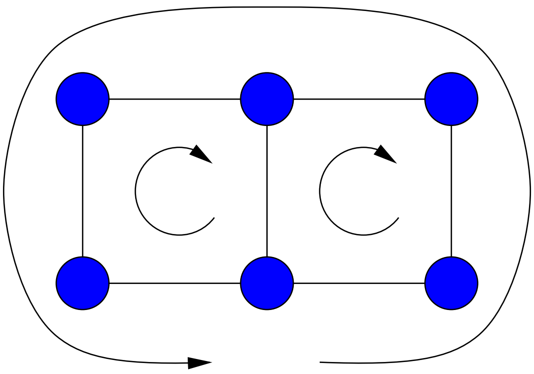

Figure B.4: A simple planar graph and its orientable cycle double cover.

the smallest legal colour in some order. Usually a good order is such that the nodes are sorted in the order of their degrees.

Algorithm 6 Greedy node colouring algorithm

let $G = (V, E)$ be an arbitrary graph, where $V = \{v_1, v_2, \ldots, v_n\}$
$c_1 \leftarrow 1$
$V' = \{v_1\}$
for $i = 2$ to n **do**
  $V' \leftarrow V' \cup \{v_i\}$
  $c_i \leftarrow \min_k \{c_1, c_2, \ldots, c_{i-1}, k\}$ is legal colouring of subgraph V'
end for

B.6 Cycle Double Cover Conjecture

Definition B.2 *A cycle double cover (CDC) of given graph is a collection of cycles such that every edge is included exactly twice.*

Definition B.3 *An orientable cycle double cover (OCDC) is such a CDC that each cycle can be given a direction and every edge is passed in both directions.*

The existence of CDC and OCDC for any bridgeless graph have been conjectured but not proven yet. For any planar graph it is easy to find OCDC (and CDC) by forming the cycles so that every face of the graph is walked in the anti-clockwise direction and one additional cycle going around the whole graph is walked in the clockwise direction (see Fig. B.4).

Appendix C

Markov Decision Processes

The theory of Markov decision processes (MDP) is fundamental to the analysis of many stochastic systems. It considers a stochastic system where decisions are made and the aim is to find an optimal strategy in some sense. In this chapter a brief introduction to the MDP theory with main results is given. A more thorough treatment can be found for example in [Tij94, Dzi97].

The contents of this appendix are briefly the following. First in Section C.1 a Markov chain is described, and then its continuous time counterpart Markov process is introduced briefly in Section C.2 with some main results important in the context of this thesis. In Section C.3 a Poisson process is presented, then in Section C.4 MDPs are presented, and finally in Section C.5 Howard's equations solving the average revenue of MDP with fixed policy are presented, followed by the description of the Policy Iteration used to find the optimal policy for given MDP.

C.1 Markov Chain

Stochastic processes can generally be classified into two categories, namely discrete time and continuous time processes [Law95, Coo81, Saa83]. Here we assume that the set of states where the system can be is finite or at least countable. A discrete time Markov process (or Markov chain) is a stochastic process where the next state at time $t+1$ depends only on its current state at time t. This *memoryless property* is fundamental to Markov processes. Formally, let X_t be the state of the process at time t, then

$$\mathrm{P}\{X_{t+1} = x | X_t = x_t, X_{t-1} = x_{t-1}, \ldots\} = \mathrm{P}\{X_{t+1} = x | X_t = x_t\}.$$

The Markov chain is defined by a transition probability matrix $\mathbf{P}$,

$$\mathbf{P} = \begin{pmatrix} p_{1,1} & p_{1,2} & \cdots \\ p_{2,1} & p_{2,2} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix},$$

where $p_{i,j}$ is the probability that the system being in state i moves to the state j in the next step. So the row sums $\sum_j p_{i,j}$ are equal to one.

The steady state probability distribution, denoted by a row vector $\boldsymbol{\pi}$, satisfies the following equation,

$$\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{P},$$

which together with the normalization requirement, $\sum_i \pi_i = 1$, defines the steady state distribution of the system.

C.2 Markov Process

The continuous time Markov process is defined similarly as Markov chains. Let $p(j,t;i,s)$ be the conditional probability that the system will be in state j at time t if it was in state i at time s. Then the system is said to be a Markov process if the Chapman-Kolmogorov equations hold:

$$p(j,t;i,s) = \sum_k p(j,t;k,u)\,p(k,u;i,s),$$

where $s < u < y$, $i,j \in \mathcal{S}$ and $\mathcal{S}$ is the set of possible states. A similar memoryless condition holds also for the continuous time Markov process, i.e. the future of the system depends only on the current state.

The infinitesimal generator or transition rate matrix $\mathbf{Q}$ is

$$\mathbf{Q} = \begin{pmatrix} q_{1,1} & q_{1,2} & \cdots \\ q_{2,1} & q_{2,2} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix},$$

where $q_{i,i} = -q_i = -\sum_{j \neq i} q_{i,j}$. It defines the transition probabilities per time unit. The state probability distribution satisfies the differential equation

$$\frac{d}{dt}\boldsymbol{\pi}(t) = \boldsymbol{\pi}(t)\mathbf{Q},$$

where $\boldsymbol{\pi}(t)$ defines the probability distribution at time t.

The steady state distribution $\boldsymbol{\pi}$ of Markov process defined by matrix $\mathbf{Q}$ can obtained from

$$\boldsymbol{\pi}\mathbf{Q} = 0.$$

The *sojourn* times in each state i are exponentially distributed with mean $\tau_i = 1/\sum_{j \neq i} q_{i,j}$.

C.3 Poisson Process

The Poisson process is probably the most used traffic (arrival) model in data communications as well as in many other areas. The Poisson process is a pure birth process with Markovian property. In this context we talk about arrivals. Briefly, the probability of an arrival during a small time interval of Δt is proportional to the length of time interval, i.e.

$$\mathrm{P}\{\text{one arrival during } \Delta t\} = \lambda \cdot \Delta t + o(t),$$

where constant λ is called arrival rate. It follows that the inter arrival times are exponentially distributed with parameter λ. This can also be expressed by saying that during a fixed length time interval t the number of arrivals N_t obeys Poisson distribution with parameter λt,

$$\mathrm{P}\{i \text{ arrivals during time } t\} = \mathrm{P}\{N_t = i\} = \frac{(\lambda t)^i}{i!} e^{-\lambda t}.$$

C.4 MDP and Policy

A Markov decision process (MDP) is a stochastic process on which a user has some control, i.e. the user can make decisions at certain points of process. The set of possible decisions depends on the current state of the system, but also on the event that occurred. Denote with $\mathcal{S}$ the system state space and with $\mathcal{K}$ the set of possible events.

The decision in each state i for each possible event k defines the policy α, and vice versa. The Markov property guarantees that the future of the system (including events) depends only on the current state of the system. In particular, given the current state, the decisions made in the past have no effect on the future. Hence, it is natural to assume that the policy is not time dependent.

The decision for the input pair (i, k) may involve (or generate):

1. a cost, e.g. blocking of type k call implies some cost
2. a transition from state i to another state j, where pair (i, k) defines the set of possible new states

Assume that the user has to make a decision in state i for certain event k. Let the set $\mathcal{A}_{i,k}$ contain all the possible new states, and if one possible decision is to stay in the same state, then also $i \in \mathcal{A}_{i,k}$. Furthermore, let each decision have a unique immediate cost $f_{i,j}(k)$, which is zero if the decision itself costs nothing. Note that $f_{i,i}(k)$ denotes the immediate cost of a decision that leaves the system in state i.

The new state j identifies the action since, if there were several actions leading to the same state j, it is clearly sufficient to consider only the most profitable among them, as the past decisions have no affect on the future. Hence, the policy α picks one state j from each $\mathcal{A}_{i,k}$, i.e. it can be written that,

$$\alpha : \mathcal{S} \times \mathcal{K} \to \mathcal{S}, \qquad \text{where } \alpha(i,k) \in \mathcal{A}_{i,k} \subseteq \mathcal{S} \qquad \forall\, i,k.$$

Once the policy α is fixed the resulting stochastic process is a Markov process (or Markov chain) X_t with some cost rate in each state (or revenue rate). The problem is to find the optimal policy α which minimizes the expected cost rate (or maximizes revenue rate). Note that for a fixed policy α the resulting Markov process (or Markov chain) no longer contain events $k \in \mathcal{K}$ explicitly.

C.4.1 Cost Model for Markov Chain

Denote the transition probabilities of a given Markov chain under policy α with $p_{ij}(\alpha)$. The steady state distribution $\pi_i(\alpha)$, defined by the transition probabilities, defines the long time average being in each state i, i.e. during N rounds, where N is large, the Markov chain resides in state i on average $N \cdot \pi_i(\alpha)$ times .

Basically the costs in a Markov chain can be placed in

1. transitions, i.e. each transition $i \to j$ has unique cost c_{ij}, or
2. visits, i.e. every visit to state i costs c_i.

Furthermore, assume that in either case the policy determines only the transition probabilities but not the fundamental costs c_{ij} or c_i. Here it is assumed that costs are deterministic, otherwise their expected values could be used. This leads to three different possible definitions for the so called immediate cost of state i (or reward of state i):

1. *visit costs*, when costs are defined per visit, i.e.
$$r_i(\alpha) = r_i = c_i. \tag{C.1}$$
2. *pre-visit costs*, the average cost from transition to state i under policy α, i.e.
$$r_i^{(-)}(\alpha) := \frac{\sum_j \pi_j(\alpha) \cdot p_{ji}(\alpha) \cdot c_{ji}}{\sum_j \pi_j(\alpha) \cdot p_{ji}(\alpha)}. \tag{C.2}$$
3. *post-visit costs*, the average cost from transition to the next state from state i under policy α, i.e.
$$r_i^{(+)}(\alpha) := \sum_j p_{ij}(\alpha) \cdot c_{ij}. \tag{C.3}$$

Note that in the case of visit costs the immediate cost of state does not depend on the policy. Nonetheless, each alternative definition attaches a certain cost for visit in each state and optimal policy minimizes the number of visits in states weighted with immediate cost. Hence, the long time average cost per round $r(\alpha)$ is

$$r(\alpha) = \sum_i \pi_i(\alpha) \cdot r_i, \qquad \text{or} \tag{C.4}$$

$$r(\alpha) = \sum_i \pi_i(\alpha) \cdot r_i^{(-)}(\alpha) = \sum_i \pi_i(\alpha) \cdot r_i^{(+)}(\alpha). \tag{C.5}$$

The number of arrivals to state i is equal to the visits to state i, and furthermore is equal to the number of departures from state i. If it holds that

$$c_i(\alpha) = \sum_j p_{i,j} \cdot c_{ij},$$

then the long time average cost per round is the same in (C.4) and (C.5). Thus, the sources of costs can be placed different ways still leading to the same optimal policy α. Placement of costs, however, leads to different relative values when solving Howard's equations and formulation of policy iteration.

C.4.2 Cost Model for Markov Processes

A Markov process is a continuous time process unlike a Markov chain. Also the costs can originate from even more numerous sources than in the case of a Markov chain. Namely,

1. *visit costs*, each visit in state i costs c_i.
2. *transition costs*, each transition $i \to j$ costs c_{ij}.
3. *cost rate*, the costs are incurred in rate r_i' while the system is in state i.

No matter how the fundamental costs are defined, it is easy to obtain equivalent (total) cost rate in state i in the following way. Let the average time spent in state i be $\tau_i = \pi_i \cdot T$. Then, the costs incurred in state i are on average

$$\tau_i r_i = \tau_i r_i' + \sum_{j \neq i} \tau_i q_{ij} c_{ij} + \tau_i q_i c_i.$$

Hence, the average cost rate in state i is

$$r_i = r_i' + \sum_{j \neq i} q_{ij} c_{ij} + q_i c_i.$$

Assuming the fundamental costs are policy independent, the equation becomes

$$r_i(\alpha) = r_i' + \sum_{j \neq i} q_{ij}(\alpha) \cdot c_{ij} + q_i(\alpha) \cdot c_i.$$

Thus, when the steady state distribution π_i of the system is obtained, the average cost rate is simply

$$r(\alpha) = \sum_i r_i(\alpha) \cdot \pi_i(\alpha).$$

C.5 Howard's Equations

Howard's equations, presented in the following sections, provide a systematic procedure to obtain the average revenue without solving the steady state probability distribution of the system.

C.5.1 Discrete Time Howard's Equations

The *relative value* (sometimes called also the relative cost) of state i, denoted with v_i, is the difference in the expected costs between a process that starts from state i and another process that starts from the equilibrium. Formally,

$$\begin{aligned} v_i &= \sum_{t=0}^{\infty} (\mathrm{E}[r_{X_t} | X_0 = i] - r) \\ &= \sum_{t=0}^{\infty} \left(\sum_{j=1}^{n} \mathrm{P}\{X_t = j | X_0 = i\} r_j - r \right) \\ &= \sum_{t=0}^{\infty} \sum_{j=1}^{n} (\mathrm{P}\{X_t = j | X_0 = i\} - \pi_j)\, r_j, \end{aligned}$$

which can be assumed to be finite, as

$$\mathrm{E}[r_{X_t} | X_0 = i] \overset{t \to \infty}{\to} r \qquad \forall\, i.$$

The difference in costs between the two processes is essentially collected during the transition phase, when the system tends towards the equilibrium from the initial state i.

The placement of the costs (pre, visit or post, equations (C.1), (C.2) and (C.3)) defines the immediate costs r_i.

The discrete time Howard's equation for state i is

$$v_i(\alpha) = r_i(\alpha) - r(\alpha) + \sum_j p_{ij}(\alpha) \cdot v_j(\alpha). \tag{C.6}$$

The formula can be explained in the following way. In the current state, before the departure, the immediate relative value is $r_i - r$. After that the system moves to state j with probability of p_{ij} and from that point onwards the incurred relative values are v_j (due to memoryless property). Taking the sum over j includes all the possible cases. Formally,

$$\begin{aligned}
v_i &= \sum_{t=0}^{\infty} (\mathrm{E}[r_{X_t}|X_0 = i] - r) \\
&= r_i - r + \sum_{t=1}^{\infty} (\mathrm{E}[r_{X_t}|X_0 = i] - r) \\
&= r_i - r + \sum_{t=1}^{\infty} \left(\left(\sum_j p_{ij} \mathrm{E}[r_{X_t}|X_1 = j] \right) - r \right) \\
&= r_i - r + \sum_j p_{ij} \sum_{t=1}^{\infty} (\mathrm{E}[r_{X_t}|X_1 = j] - r) \\
&= r_i - r + \sum_j p_{ij} v_j .
\end{aligned}$$

Hence, there are N linear equations and $N+1$ unknown variables. Any of the relative values v_i can be fixed to be 0 (or any other finite value). If a constant C is added to each relative value, they still satisfy equation (C.6). Hence, a constant offset in relative costs $\{v_i\}$ has no effect on r or the optimal policy. Once one of the relative values is fixed we are left with N unknown variables so that the interesting quantity, the average relative cost r, can be obtained.

C.5.2 Continuous Time Howard's Equations

Howard's equations can also be used with continuous time processes. Denote the relative costs again with v_i, i.e.

$$v_i = \lim_{T \to \infty} \int_0^T (\mathrm{E}[r_{X_t}|X_0 = i] - r) \; dt,$$

where r is the average cost rate in the long run and $\mathrm{E}[r_{X_t}|X_0 = i]$ is the expected cost rate at time t when the process starts initially from state i (see Fig. C.1). We can assume that the above limit exists and is finite for each i. If actual costs are defined by transitions or visits in states, it is straightforward to combine them to cost rates and obtain an equivalent Markov process with cost rates as presented in previous section.

We proceed by considering the embedded Markov chain. The transition probabilities of the embedded Markov chain are

$$p_{ij} = \begin{cases} \frac{q_{ij}}{q_i}, & \text{when } i \neq j, \\ 0, & \text{when } i = j. \end{cases}$$

Assume, that while the system is in state i the rate at which costs are incurred is r_i. Then the average incurred cost while staying in the current state, i.e. the

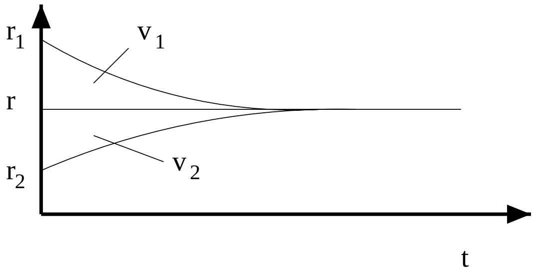

Figure C.1: Illustration of relative costs v_i of a continuous time MDP.

immediate cost in embedded Markov chain, is

$$\frac{r_i - r}{q_i}.$$

Substituting these to (C.6), gives

$$v_i = \frac{r_i - r}{q_i} + \sum_{j \neq i} \frac{q_{ij}}{q_i} v_j.$$

Recalling that $q_i = \sum_{j \neq i} q_{ij}$, we get the continuous time Howard's equations:

$$r_i - r + \sum_{j \neq i} q_{ij}(v_j - v_i) = 0, \qquad \forall\, i \tag{C.7}$$

Since $q_{ii} = -\sum_{j \neq i} q_{ij}$, the following alternative formulation for Howard's equations is obtained,

$$r_i - r + \sum_{j} q_{ij} v_j = 0, \qquad \forall\, i \tag{C.8}$$

As in the discrete time case, an arbitrary relative cost v_i can be fixed, after which the rest of the relative costs and the average cost rate r can be solved.

Equation (C.8) can be explained in the following way. The difference in income rates in the current state equals to $r_i - r$, and the summation gives the transition rates to other states weighted with the appropriate relative values.

C.6 Policy Iteration

Next a systematic procedure to obtain the optimal policy iteratively is presented. The procedure starts from an arbitrary initial policy α and in each round a new better policy is obtained by using the so called policy improvement step [Dzi97, Tij94].

Recall that the relative values v_i (presented in the context of Howard's equations) represent the difference in the expected future costs for the system starting from certain state i rather than from the equilibrium. The decisions the

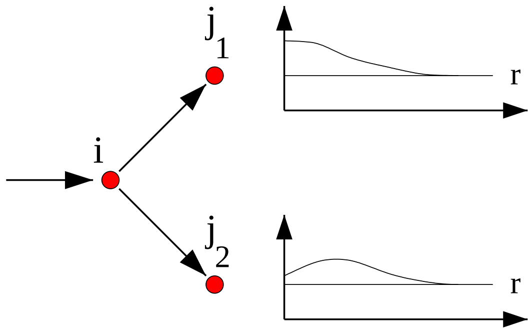

Figure C.2: The two possible decision depicted. After the decision is made (and paid the immediate cost) the future (relative) costs, assuming policy $\boldsymbol{\alpha}$, are given by $\boldsymbol{v_{j_1}}$ and $\boldsymbol{v_{j_2}}$ accordingly.

user makes defines the policy. Due to the memoryless property the policy can also be assumed to be time independent, i.e. at the same state the policy makes always the same decision.

As presented before, the policy α defines decisions in each state i for every event k. Furthermore, the decision made may generate some cost and a transition. The set $\mathcal{A}_{i,k}$ contains possible new states when type k event occurs in state i. Note that it contains also the current state i if one of the decisions is to stay in the same state. As stated before, a policy is explicitly defined by the state the system is after the decision. Denote with $f_{i,j}(k)$ the cost of the decision to move to state j when type k event occurs in state i. This corresponds to the transition cost c_{ij} presented in C.4.2 with an additional parameter k. Formally, the mapping

$$\alpha : \mathcal{S} \times \mathcal{K} \to \mathcal{S},$$

together with constraint $\alpha(i,k) \in \mathcal{A}_{i,k}$, defines a policy.

C.6.1 Policy Iteration I

Assume that we have an arbitrary policy α. While being in state i an event k occurs, so that a decision or an action should be made. The action $a_{i,k}$ to be taken should be the one which minimizes the expected future costs. Assume that, once the action is taken, the system reverts back to the standard policy α. Then the optimal action is clearly

$$a_{i,k} = \underset{j \in \mathcal{A}_{i,k}}{\arg\min} \left\{ f_{i,j}(k) + v_j(\alpha) \right\}, \qquad \forall\, i, k, \tag{C.9}$$

The equation defines the action for each possible state i and event k, i.e. a new policy α'. For the original policy α the expected relative future costs are known. So, taking a minimum over all the possible actions, a better or at least equal policy is obtained. Repeating the iteration the optimal policy will be finally reached. The policy iteration algorithm is presented in Algorithm 7. Note that the same algorithm holds for the discrete and continuous MDP cases.

Algorithm 7 Policy iteration

Let α be an arbitrary policy
loop
  Solve Howard's equations for the current policy $\alpha \Rightarrow$ relative values v_i and the average revenue rate r for the current policy
  if Average revenue rate r did not improve **then**
    break
  end if
  Determine a new policy α', for each state from: $\underset{j \in \mathcal{A}_{i,k}}{\arg\min} \{f_{i,j} + v_j\}$,
  which defines the action to be taken in each state i when type k event occurs.
  $\alpha \leftarrow \alpha'$
end loop

C.6.2 Policy Iteration II

The policy iteration step can be expressed in another way, too. Namely, consider that the alternative policy is used until the next transition occurs. Then the new policy is defined by the following equation:

$$\underset{\alpha'}{\arg\min} \left\{ r_i(\alpha') + \sum_j q_{ij}(\alpha') \cdot v_j(\alpha) \right\}, \qquad \forall\, i, \tag{C.10}$$

which is similar to Howard's equation (C.8) except that the average cost rate r, being just a constant factor, is dropped, and the cost rate r_i and the transition intensity matrix $\mathbf{Q}$ are obtained for different policy than the relative costs v_i. By taking arg min over all the possible "temporary" policies α', we get a new and never worse policy. Again, the policy improvement step is repeated until the iteration step does not change the policy (or the average cost r does not decrease if there are two or more optimal policies).

Appendix D

Network Simulator

Network simulator program, “Verkko”, is used to simulate the routing and wavelength assignment problems in fully optical networks. The program is written from scratch in Ansi C language, and the source code currently contains about 10000 lines.

The program is command line driven and uses auxiliary files which define the structure of the physical network as well as the offered traffic. Based on this information the program simulates the behaviour of the given optical network and reports blocking probabilities and cumulative costs per traffic class. Several RWA algorithms are built in the simulator and adding a new algorithm is a quite straightforward task. Special attention was paid to the data structures and the memory handling routines in order to optimize the simulation times[1].

D.1 Routes

Both static and dynamic traffic cases use pre-calculated routes, i.e. for each node pair a set of route candidates are stored in advance in linked lists [Knu68]. A linked list is a dynamic data structure which suits well to describe routes of different lengths. Exception to this rule is the adaptive unconstrained routing (AUR) algorithms, which determine the route dynamically based on the current state of the network. The routing parameters define what kind of routes are accepted and how many. The possible parameters are the following (each is of restrictive type):

- Δl, this parameter limits the number of extra (optical) hops the route can make when compared to shortest possible route. If Δl is zero only the shortest paths are included.

[1] An important factor when using the first policy iteration.

- *rmax*, this parameter limits the total number of route candidates. If more than *rmax* routes, then the shortest paths are included.

- *maxtest*, this parameter defines the maximum alternative routes actually checked when the first policy iteration is applied (see Section 4.5).

D.2 Static Traffic

The configuration of the network is split to two subproblems. First the routes ares fixed and then the wavelengths are assigned. The problem can be divided to two categories:

- Single-fibre case:
  - Graph node colouring approach, many algorithms presented in the literature implemented.
  - Handles the WIXC case appropriately by splitting the route to sub routes and assigning a wavelength to each hop independently
- Multi-fibre case
  - Arbitrary number of fibres in any link
  - WIXC case as in the single fibre case, i.e. each optical hop is assigned separately.
  - Only one algorithm implemented: simple greedy algorithm assigns wavelengths after routing

D.3 Dynamic Traffic

- Calls arrive with rate λ_i (Poisson) and call durations follow the exponential distribution with mean $1/\mu_i$, where i is the traffic class.

- A flag defines whether the duration of incoming connection request is known or not.

- Supports saving/loading of arrivals to/from an external file

- Currently implemented algorithms are *basic*, *porder*, *pcolor*, *pcolor*, *ll*, *iteration* and *reroute*.
  - *basic*, tries every wavelength with every route in that order
  - *porder*, tries every route with every wavelength in that order
  - *pcolor*, tries every route in order of the most used wavelengths first

- *lpcolor*, minimizes primarily the length of the route, and secondarily uses the most used wavelength
- *reroute*, tries first using basic algorithm and if that fails all connections are rerouted

- Also adaptive versions (AUR): *pack*, *spread*, *random*, *exhaustive* and *fixed* order

- Notes:
  - *basic*, *porder* and *pcolor* algorithms do not handle WIXC case yet
  - *reroute* algorithm uses static traffic routines and has no limitations

D.4 Verification of Simulator Software

The simulator sources consist of many thousands lines of C-code, which means that there is very likely more than one bug hiding. Hence, during the program development it is important to frequently run tests to find possible problems to be able to correct them immediately.

The tests run can be divided into the following categories:

- C-compiler warnings must be fixed, this merely a sign of a good programming style. With GNU ([Sta84]) gcc-compiler a *-Wall* option gives satisfactory reports about likely errors.

- Test runs with simple cases that can be solved analytically. As the general problem is clearly impossible to be solved analytically, some simple cases can still be calculated and simulated results can be compared to them. For example, a two node network behaves exactly like a normal $M/M/m/m$ queue, where m is number of channels available on link(s).

- The first policy iteration approach parameters can be estimated by tweaking the costs of rejected call to zero. Then every call should be rejected, as it costs nothing to reject it. However, due to the simulation noise, wrong decisions are sometimes made.

- Simulation runs with different parameter sets that should give the same results (scaling time, dividing the iteration to n sub periods etc.).

- Comparisons with results from the published papers. Both published and own results contain simulation noise, but the order of the results should match.

Listing D.1: Example sections defining the physical network.

```
#NODES
Hki       0      0     o
Espoo    -0.1    0     o
Vantaa    0      0.1   o
Turku    -0.8    0     o
Vaasa    -1.2    2     o
Tre      -0.5    0.8   o
Jkl       0.3    1.7   o
Lpr       1      0.9   o
Joensuu   1.2    2.5   o
Kuopio    0.7    2.5   o
Oulu      0      3.5   o
#END

#LINKS
Hki      Espoo     1
Hki      Vantaa    1
Espoo    Vantaa    1
Espoo    Turku     1
Vantaa   Tre       1
Vantaa   Lpr       1
Turku    Vaasa     1
Tre      Jkl       1
Tre      Turku     1
Lpr      Joensuu   1
Kuopio   Joensuu   1
Jkl      Kuopio    1
Kuopio   Oulu      1
Oulu     Vaasa     1
#END
```

D.5 Network File Formats

The network file consists of two sections. The first part *NODES* defines the network nodes and the second section *LINKS* defines the links between nodes. Each line of *NODES* section contains one entry:

```
<name>    <x coordinate>  <y coordinate>  <type>
```

The coordinates can be used to present graphical picture of the network, but they have no effect to actual problem nor simulations. The type of the node can be either

o normal, no wavelength conversions, or

x wavelength conversion capable node.

Each section is ended with *#END* keyword. The second section *LINKS* defines the links between the network nodes:

```
<node 1>  <node 2>   <number of fibres>
```

An example network definition is presented in Listing D.1.

Listing D.2: Example section defining the traffic (2 classes).

```
#TRAFFIC 1

#POISSON normal
Hki      Espoo   2.0   1.0   1.0
Hki      Tre     1.0   1.0   1.0
Espoo    Tre     1.0   1.0   1.0
#END

#POISSON known_end
Hki      Espoo   0.5   1.0   2.0
Espoo    Tre     0.2   1.0   2.0
#END

#END
```

D.5.1 The Traffic File Format

The traffic file defines the offered traffic in the dynamic traffic case. If it is not specified, uniform traffic is assumed among all possible node pairs. Each traffic class defines a node pair, arrival intensity (Poisson traffic), average connection duration (exponential distribution) and a weight factor (cost of lost call).

The file starts with the header:

```
#TRAFFIC 1
```

The word *TRAFFIC* refers to the traffic definitions and *1* is the version number. After that comes the sections of different traffic types.

```
#POISSON normal
#POISSON known_end
```

Currently the only traffic type is Poisson traffic with either *normal* type which means the ending time of call is unknown, or *known_end* type which means that once a connection request arrives its duration becomes also known. The duration is still drawn from the exponential distribution.

After header each line contains data for one traffic class:

```
<node 1>  <node 2>   <lambda>   <mu>   <weight>
```

Note that connections are assumed to be bidirectional always. The end of each section is marked with *#END* keyword. An example traffic section is presented in Listing D.2.

Bibliography

[AH77] Kenneth Appel and Wolfgang Haken. The solution of the four-color-map problem. *Scientific American*, 237(4):108–121, October 1977.

[Ahl00] Ulf Ahlfors. *Dynamic Traffic Control in Multiservice Networks*. PhD thesis, Lund Institute of Technology, Lund University, 2000.

[AJ83] David Applegate and David Johnson. dfmax.c. `ftp://dimacs.rutgers.edu/pub/challenge/graph/solvers/`, 1983. Semi-Exhaustive Greedy Independent Set, from Matula and Johri.

[Bar98] S. Baroni. *Routing and wavelength allocation in WDM optical networks*. PhD thesis, University College London, May 1998.

[BB97] Stefano Baroni and Polina Bayvel. Wavelength requirements in arbitrarily connected wavelength-routed optical networks. *IEEE Journal of Lightwave Technology*, 15(2):242–251, February 1997.

[BG92] D. Bertsekas and R. Gallager. *Data Networks*. Prentice-Hall, 1992.

[BGD01] David J. Bishop, C. Randy Giles, and Saswato R. Das. The rise of optical switching. *Scientific American*, 284(1):74–79, January 2001.

[BH96] R. A. Barry and P. A. Humblet. Models for blocking probability in all-optical networks with and without wavelength exchangers. *IEEE J. Select. Areas Commun.*, 14(5):858–867, June 1996.

[Bir96] Alexander Birman. Computing approximate blocking probabilities for a class of all-optical networks. *IEEE J. Select. Areas Commun.*, 14(5):852–857, June 1996.

[Blu01] Daniel J. Blumenthal. Routing packets with light. *Scientific American*, 284(1):80–83, January 2001.

[BM76] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications*. The MacMillan Press, 1976.

[BM96] Dhritiman Banaree and Biswanath Mukherjee. A practical approach for routing and wavelength assignment in large wavelength-routed optical networks. *IEEE J. Select. Areas Commun.*, 14(5):903–908, June 1996.

[Bré79] D. Brélaz. New methods to colour the vertices of a graph. *In Communications of the ACM*, 22(4), 1979.

[BRM00] Paul Bonenfant and Antonio Rodriguez-Moral. Optical data networking. *IEEE Commun. Mag.*, 38(2):63–70, February 2000.

[BSSLB95] Krishna Bala, Thomas E. Stern, David Simchi-Levu, and Kavita Bala. Routing in a linear lightwave network. *IEEE/ACM Transactions on Networking*, 3(4):459–469, June 1995.

[Coo81] Robert B. Cooper. *Introduction to Queueing Theory*. Elsevier Science Publishing, 1981.

[Dud89] Richard M. Dudley. *Real Analysis and Probability*. Wadsworth & Brooks / Cole, 1989.

[Dzi97] Zbigniew Dziong. *ATM Network resource management*. McGraw-Hill, 1997.

[EM00a] Jaafar M. H. Elmirghani and Hussein T. Mouftah. All-optical wavelength conversion: Technologies and applications in DWDM networks. *IEEE Commun. Mag.*, 38(3):86–92, March 2000.

[EM00b] Jaafar M. H. Elmirghani and Hussein T. Mouftah. Technologies and architectures for scalable dynamic dense WDM networks. *IEEE Commun. Mag.*, 38(2):58–66, February 2000.

[GDW00] Nasir Ghani, Sudhir Dixit, and Ti-Shang Wang. On IP-over-WDM integration. *IEEE Commun. Mag.*, 38(2):72–84, February 2000.

[Gha00] Nasir Ghani. Lambda-labeling: A framework for IP-over-WDM using MPLS. *Optical Networks Magazine*, 1(2):45–58, April 2000.

[GSM97] M. Garnot, M. Sotom, and F. Masetti. Routing strategies for optical paths in WDM networks. *IEEE International Conference on Communications 1997*, pages 422–446, 1997.

[HdW87] A. Hertz and D. de Werra. Using tabu search techniques for graph coloring. *Computing*, 39:345–351, 1987.

[HMM99] H. Harai, M. Murata, and H. Miyahara. Heuristic algorithm for allocation of wavelength convertible nodes and routing coordination in all-optical networks. *IEEE J. Lightwave Technol.*, 17(4), April 1999.

[Hui95] Christian Huitema. *Routing in the Internet*. Prentice-Hall Inc., 1995.

[HV98] Esa Hyytiä and Jorma Virtamo. Wavelength assignment and routing in WDM networks. In *NTS 14, Lyngby, Denmark*, pages 31–40, August 1998.

[HV99] Esa Hyytiä and Jorma Virtamo. Wavelength assignment in multifibre WDM-networks. Technical Report 99(04), COST257, 1999.

[HV00a] Esa Hyytiä and Jorma Virtamo. Dynamic routing and wavelength assignment using first policy iteration. In *ISCC'2000, the Fifth IEEE Symposium on Computers and Communications, Antibes, Juan les Pins, France,* pages 146–151, July 2000.

[HV00b] Esa Hyytiä and Jorma Virtamo. Dynamic routing and wavelength assignment using first policy iteration, inhomogeneous traffic case. In *P&QNet2000, the International Conference on Performance and QoS of Next Generation Networking, Nagoya, Japan,* pages 301–316, November 2000.

[ITU98] ITU-T. *G.692, Optical interfaces for multichannel systems with optical amplifiers,* October 1998.

[KA96] Milan Kovačević and Anthony Acampora. Benefits of wavelength translation in all-optical clear-channel networks. *IEEE J. Select. Areas Commun.*, 14(5):868–880, June 1996.

[KA98] Ezhan Karasan and Ender Ayanoglu. Effects of wavelength routing and selection algorithms on wavelength conversion gain in WDM optical networks. *IEEE/ACM Trans. Networking*, 6(2):186–196, April 1998.

[Knu68] Donald E. Knuth. *The Art of Computer Progrmming: Volume 1, Fundamental Algorithms.* Addison-Wesley, 1968.

[Kri90] K. R. Krishnan. Markov decision algorithms for dynamic routing. *IEEE Commun. Mag.*, 28(10):66–69, October 1990.

[Kri91] K. R. Krishnan. Markov decision algorithms for dynamic routing. *GlobeCom'91*, 2:1346–1350, 1991.

[KS99] Donald L. Kreher and Douglas R. Stinson. *Combinatorial Algorithms: generation, enumeration and search.* CRC Press series on discrete mathematics and its applications. CRC Press, 1999.

[Law95] Gregory F. Lawler. *Introduction to Stochastic Processes.* Chapman & Hall Probability Series. Chapman & Hall, 1995.

[MA98] A. Mokhtar and M. Azizoğlu. Adaptive wavelength routing in all-optical networks. *IEEE/ACM Trans. Networking*, 6(2):197–206, April 1998.

[MBRM96] Biswanath Mukherjee, Dhritiman Banaree, S. Ramamurthy, and Amarnath Mukherjee. Some principles for designing a wide-area WDM optical network. *IEEE/ACM Trans. Networking*, 4(5):684–696, October 1996.

[Mit76] John Mitchem. On various algorithms for estimating the chromatic number of a graph. *The Computer Journal*, 19(2):182–183, 1976.

[MM99] G. Mohan and S.R. Murthy. A time optimal wavelength rerouting algorithm for dynamic trafffic in WDM networks. *IEEE J. Lightwave Technol.*, 17(3):406–417, March 1999.

[Muk97] Biswanath Mukherjee. *Optical Communication Networks*. McGraw-Hill series on computer communications. McGraw-Hill, 1997.

[PMMB00] A. Pattavina, M. Martinelli, G. Maier, and P. Boffi. Techniques and technologies towards all-optical switching. *Optical Networks Magazine*, 1(2):75–93, April 2000.

[Ree95] Colin R. Reeves, editor. *Modern Heuristic Techniques for Combinatorial Problems*. Advanced Topics in Computer Science Series. McGraw-Hill, 1995.

[Ros00] Sheldon M. Ross. *Introduction to probability models*. Academic Press, 7th edition, 2000.

[RRP99] Sridar Ramesh, George N. Rouskas, and Harry G. Perros. Computing blocking probabilities in multi-class wavelength routing networks. Technical Report TR-99-08, Department of Computer Science, College of Engineering, North Carolina State University, May 1999. http://www.csc.ncsu.edu/research/techreports/README.html.

[RS95] Rajiv Ramaswami and Kumar N. Sivarajan. Routing and wavelength assignment in all-optical networks. *IEEE/ACM Trans. Networking*, 3(5):489–500, October 1995.

[RS96] Rajiv Ramaswami and Kumar N. Sivarajan. Design of logical topologies for wavelength-routed optical networks. *IEEE J. Select. Areas Commun.*, 14(5):840–851, June 1996.

[RS98] Rajiv Ramaswami and Kumar N. Sivarajan. *Optical Networks, A Practical Perspective*. Morgan Kaufmann Series in Networking. Morgan Kaufmann Publishers, 1998.

[RSORS96] V.J Rayward-Smith, I.H. Osman, C.R. Reeves, and G.D. Smith. *Modern Heuristic Search Methods*. John Wiley & Sons, 1996.

[Rum00] Hannu Rummukainen. On approximate markov control of multiservice telecommunication links. Master's thesis, Helsinki University of Technology, 2000.

[Saa83] Thomas L. Saaty. *Elements of Queueing Theory With Applications*. Dover Publications, 1983.

[SB97] Suresh Subramaniam and Richard A. Barry. Wavelength assignment in fixed routing WDM networks. *IEEE International Conference on Communications 1997*, pages 406–410, 1997.

[SB99] Thomas E. Stern and Krishna Bala. *Multiwavelength Optical Networks: a layered approach*. Addison Wesley, 1999.

[SK77] Thomas L. Saaty and Paul C. Kainen. *The Four-Color Problem, Assaults and Conquest*. McGraw-Hill, 1977.

[Sta84] Richard Stallman. The GNU project. `http://www.gnu.org/`, 1984.

[Sti01] Gary Stix. The triumph of the light. *Scientific American*, 284(1):68–73, January 2001.

[Tah92] Hamdy. A. Taha. *Operations Research, An Introduction*. Macmillan publishing company, 5th edition, 1992.

[Tij94] Henk C. Tijms. *Stochastic Models, An Algorithmic Approach*. John Wiley & Sons Ltd, 1994.

[TP95] T.K. Tan and J.K. Pollard. Determination of minimum number of wavelength required for all-optical WDM networks using graph colouring. *Electronic letters*, 31(22):1895–1897, 1995.

[Wei97] Mark Allen Weiss. *Data Structures and Algorithm Analysis in C*. Addison-Wesley, 1997.

[Wil91] David Williams. *Probability with Martingales*. Cambridge University Press, 1991.

[Wil97] A.E. Willner. Mining the optical bandwidth for a terabit per second. *IEEE Spectrum*, 34(3):32–41, April 1997.

[YMD00] Shun Yao, Biswanath Mukherjee, and Sudhir Dixit. Advances in photonic packet switching: An overview. *IEEE Commun. Mag.*, 38(2):84–94, February 2000.

[ZA95] Zhensheng Zhang and Anthony S. Acampora. A heuristic wavelength assignment algorithm for multihop WDM networks with wavelength routing and wavelength re-use. *IEEE/ACM Trans. Networking*, 3(3):281–288, June 1995.

[ZAA+97] Lisa Zhang, Matthew Andrews, William Aieollo, Sandeep Bhatt, and K. R. Krishnan. A performance comparison of competive online routing and state-dependent routing. *GlobeCom'97*, 3:1813–1819, 1997.

[ZRP98a] Yohong Zhu, George N. Rouskas, and Harry G. Perros. Blocking in wavelength routing networks, part I: The single path case. Technical Report TR-98-02, Department of Computer Science, College of Engineering, North Carolina State University, February 1998. http://www.csc.ncsu.edu/research/tech-reports/README.html.

[ZRP98b] Yohong Zhu, George N. Rouskas, and Harry G. Perros. Blocking in wavelength routing networks, part II: Mesh topologies. Technical Report TR-98-10, Department of Computer

Science, College of Engineering, North Carolina State University, October 1998. http://www.csc.ncsu.edu/research/tech-reports/README.html.

[Öst99] Patric Östergård. wclique.c - a maximum-weight clique program. `http://www.tcs.hut.fi/~pat/wclique.html`, January 1999.

Index