International Series in
Operations Research & Management Science

Richard J. Boucherie
Nico M. van Dijk *Editors*

Markov Decision Processes in Practice

Springer

International Series in Operations Research & Management Science

Volume 248

Richard J. Boucherie • Nico M. van Dijk
Editors

Markov Decision Processes in Practice

Springer

Editors
Richard J. Boucherie
Stochastic Operations Research
University of Twente
Enschede, The Netherlands

Nico M. van Dijk
Stochastic Operations Research
University of Twente
Enschede, The Netherlands

ISSN 0884-8289 ISSN 2214-7934 (electronic)
International Series in Operations Research & Management Science
ISBN 978-3-319-47764-0 ISBN 978-3-319-47766-4 (eBook)
DOI 10.1007/978-3-319-47766-4

Library of Congress Control Number: 2017932096

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

To

Carla,
Fabian, Daphne, Deirdre, and Daniël –
Thanks for being there in difficult times,
Richard

P. Dorreboom and his daughter –
for coping with my passions,
Nico

Foreword

I had the pleasure of serving as the series editor of this series over its first 20 years (from 1993 through October, 2013). One of the special pleasures of this work was the opportunity to become better acquainted with many of the leading researchers in our field and to learn more about their research. This was especially true in the case of Nico M. van Dijk, who became a friend and overnight guest in our home. I then was delighted when Nico and his colleague, Richard J. Boucherie, agreed to be the editors of a handbook, *Queueing Networks: A Fundamental Approach*, that was published in 2010 as Vol. 154 in this series. This outstanding volume succeeded in defining the current state of the art in this important area.

Because of both its elegance and its great application potential, Markov decision processes have been one of my favorite areas of operations research. A full chapter (Chap. 19 in the current tenth edition) is devoted to this topic in my textbook (coauthored by the late Gerald J. Lieberman), *Introduction to Operations Research*. However, I have long been frustrated by the sparsity of publications that describe applications of Markov decision processes. This was less true about 30 years ago when D.J. White published his seminal papers on such *real* applications in *Interfaces* (see the November–December 1985 and September–October 1988 issues). Unfortunately, relatively few papers or books since then have delved much into such applications. (One of these few publications is the 2002 book edited by Eugene Feinberg and Adam Shwartz, *Handbook of Markov Decision Processes: Methods and Applications*, which is Vol. 40 in this series.)

Given the sparse literature in this important area, I was particularly delighted when the outstanding team of Nico M. van Dijk and Richard J. Boucherie accepted my invitation to be the editors of this exciting new book that focuses on Markov decision processes in practice. One of my last acts as the series editor was to work with these coeditors and the publisher in shepherding the book proposal through the process of providing the contract for its publication. I feel that this book may prove

to be one of the most important books in the series because it sheds so much light on the great application potential of Markov decision processes. This hopefully will lead to a renaissance in applying this powerful technique to numerous *real* problems.

Stanford University
July 2016

Frederick S. Hillier

Preface

It is over 30 years ago since D.J. White started his series of surveys on practical applications of Markov decision processes (MDP),[1,2,3] over 20 years after the phenomenal book by Martin Puterman on the theory of MDP,[4] and over 10 years since Eugene A. Feinberg and Adam Shwartz published their *Handbook of Markov Decision Processes: Methods and Applications*.[5] In the past decades, the practical development of MDP seemed to have come to a halt with the general perception that MDP is computationally prohibitive. Accordingly, MDP is deemed unrealistic and is out of scope for many operations research practitioners. In addition, MDP is hampered by its notational complications and its conceptual complexity. As a result, MDP is often only briefly covered in introductory operations research textbooks and courses. Recently developed approximation techniques supported by vastly increased numerical power have tackled part of the computational problems; see, e.g., Chaps. 2 and 3 of this handbook and the references therein. This handbook shows that a revival of MDP for practical purposes is justified for several reasons:

1. First and above all, the present-day numerical capabilities have enabled MDP to be invoked for real-life applications.
2. MDP allows to develop and formally support approximate and simple practical decision rules.
3. Last but not least, MDP's probabilistic modeling of practical problems is a skill if not art by itself.

[1] D.J. White. Real applications of Markov decision processes. *Interfaces*, 15:73–83, 1985.

[2] D.J. White. Further real applications of Markov decision processes. *Interfaces*, 18:55–61, 1988.

[3] D.J. White. A Survey of Applications of Markov Decision Processes. *Journal of the Operational Research Society*, 44:1073–1096, 1993.

[4] Martin Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, 1994.

[5] Eugene A. Feinberg and Adam Shwartz, editors. *Handbook of Markov Decision Processes: Methods and Applications*. Kluwer, 2002.

This handbook *Markov Decision Processes in Practice* aims *to show the power of classical MDP for real-life applications and optimization*. The handbook is structured as follows:

Part I: General Theory
Part II: Healthcare
Part III: Transportation
Part IV: Production
Part V: Communications
Part VI: Financial Modeling

The chapters of Part I are devoted to *the state-of-the-art theoretical foundation of MDP*, including approximate methods such as policy improvement, successive approximation and infinite state spaces as well as an instructive chapter on approximate dynamic programming. Parts II–VI contain a collection of *state-of-the-art applications in which MDP was key to the solution approach* in a non-exhaustive selection of application areas. The application-oriented chapters have the following structure:

- Problem description
- MDP formulation
- MDP solution approach
- Numerical and practical results
- Evaluation of the MDP approach used

Next to the MDP formulation and justification, most chapters contain numerical results and a real-life validation or implementation of the results. Some of the chapters are based on previously published results, some are expanding on earlier work, and some contain new research. All chapters are thoroughly reviewed. To facilitate comparison of the results offered in different chapters, several chapters contain an appendix with notation or a transformation of their notation to the basic notation provided in Appendix A. Appendix B contains a compact overview of all chapters listing discrete or continuous modeling aspects and the optimization criteria used in different chapters.

The outline of these six parts is provided below.

Part I: General Theory

This part contains the following chapters:

Chapter 1: One-Step Improvement Ideas and Computational Aspects
Chapter 2: Value Function Approximation in Complex Queueing systems
Chapter 3: Approximate Dynamic Programming by Practical Examples
Chapter 4: Server Optimization of Infinite Queueing Systems
Chapter 5: Structures of Optimal Policies in MDP with Unbounded Jumps: The State of Our Art

The first chapter, by H.C. Tijms, presents a survey of the basic concepts underlying computational approaches for MDP. Focus is on the basic principle of policy improvement, the design of a single good improvement step, and one-stage-look-ahead rules, to, e.g., generate the best control rule for the specific problem of interest, for decomposition results or parameterization, and to develop a heuristic or tailor-made rule. Several intriguing queueing examples are included, e.g., with dynamic routing to parallel queues.

In the second chapter, by S. Bhulai, using one-step policy improvement is brought down to the essence of understanding and evaluating the relative value function of simple systems that can be used in the control of more complicated systems. First, the essence of this relative value function is nicely clarified by standard birth death M/M/s queueing systems. Next, a number of approximations for the relative value function are provided and applied to more complex queueing systems such as for dynamic routing in real-life multiskilled call centers.

Chapter 3, by Martijn Mes and Arturo Pérez Rivera, continues the approximation approach and presents approximate dynamic programming (ADP) as a powerful technique to solve large-scale discrete-time multistage stochastic control problems. Rather than a more fundamental approach as, for example, can be found in the excellent book of Warren B. Powell,[6] this chapter illustrates the basic principles of ADP via three different practical examples: the nomadic trucker, freight consolidation, and tactical planning in healthcare.

The special but quite natural complication of infinite state spaces within MDP is given special attention in two consecutive chapters. First, in Chap. 4, by András Mészáros and Miklós Telek, the regular structure of several Markovian models is exploited to decompose an infinite transition matrix in a controllable and uncontrollable part, which allows a reduction of the unsolvable infinite MDP into a numerically solvable one. The approach is illustrated via queueing systems with parallel servers and a computer system with power saving mode and, in a more theoretical setting, for birth-death and quasi-birth-death models.

Next, in Chap. 5, by Herman Blok and Floske Spieksma, emphasis is on structural properties of infinite MDPs with unbounded jumps. Illustrated via a running example, the natural question is addressed, how structural properties of the optimal policy are preserved under truncation or perturbation of the MDP. In particular, smoothed rate truncation (SRT) is discussed, and a roadmap is provided for preserving structural properties.

[6] Warren B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley Series in Probability and Statistics, 2011.

Part II: Healthcare

Healthcare is the largest industry in the Western world. The number of operations research practitioners in healthcare is steadily growing to tackle planning, scheduling, and decision problems. In line with this growth, in recent years, MDPs have found important applications in healthcare in the context of prevention, screening, and treatment of diseases but also in developing appointment schedules and inventory management. The following chapters contain a selection of topics:

Chapter 6: Markov Decision Processes for Screening and Treatment of Chronic Diseases
Chapter 7: Stratified Breast Cancer Follow-Up Using a Partially Observable MDP
Chapter 8: Advance Patient Appointment Scheduling
Chapter 9: Optimal Ambulance Dispatching
Chapter 10: Blood Platelet Inventory Management

Chapter 6, by Lauren N. Steimle and Brian T. Denton, provides a review of MDPs and partially observable MDPs (POMDPs) in medical decision making and a tutorial about how to formulate and solve healthcare problems with particular focus on chronic diseases. The approach is illustrated via two examples: an MDP model for optimal control of drug treatment decisions for managing the risk of heart disease and stroke in patients with type 2 diabetes and a POMDP model for optimal design of biomarker-based screening policies in the context of prostate cancer.

In Chap. 7, by J.W.M. Otten, A. Witteveen, I.M.H. Vliegen, S. Siesling, J.B. Timmer, and M.J. IJzerman, the POMDP approach is used to optimally allocate resources in a follow-up screening policy that maximizes the total expected number of quality-adjusted life years (QALYs) for women with breast cancer. Using data from the Netherlands Cancer Registry, for three risk categories based on differentiation of the primary tumor, the POMDP approach suggests a slightly more intensive follow-up for patients with a high risk for and poorly differentiated tumor and a less intensive schedule for the other risk groups.

In Chap. 8, by Antoine Sauré and Martin L. Puterman, the linear programming approach to ADP is used to solve advance patient appointment scheduling problems, which are problems typically intractable using standard solution techniques. This chapter provides a systematic way of identifying effective booking guidelines for advance patient appointment scheduling problems. The results are applied to CT scan appointment scheduling and radiation therapy treatment scheduling.

Chapter 9, by C.J. Jagtenberg, S. Bhulai, and R.D. van der Mei, considers the ambulance dispatch problem, in which one must decide which ambulance to send to an incident in real time. This chapter develops a computationally tractable MDP that captures not only the number of idle ambulances but also the future incident location and develops an ambulance dispatching heuristic that is shown to reduce the fraction of late arrivals by 13% compared to the "closest idle" benchmark policy for the Dutch region Flevoland.

Chapter 10, by Rene Haijema, Nico M. van Dijk, and Jan van der Wal, considers the blood platelet inventory problem that is of vital importance for patients' sur-

vival, since platelets have a limited lifetime after being donated and lives may be at risk when no compatible blood platelets are available for transfusion, for example, during surgery. This chapter develops a combined MDP and simulation approach to minimize the blood platelet outdating percentage taking into account special production interruptions due to, e.g., Christmas and Easter holidays.

Part III: Transportation

Transportation science is known as a vast scientific field by itself for both the public (e.g., plane, train, or bus) and private modes of transportation. Well-known research areas include revenue management, pricing, air traffic control, train scheduling, and crew scheduling. This part contains only a small selection of topics to illustrate the possible fruitful use of MDP modeling within this field, ranging from macro-level to micro-level and from public transportation to private transportation. It contains the following chapters:

Chapter 11: Stochastic Dynamic Programming for Noise Load Management
Chapter 12: Allocation in a Vertical Rotary Car Park
Chapter 13: Dynamic Control of Traffic Lights
Chapter 14: Smart Charging of Electric Vehicles

Chapter 11, by T.R. Meerburg, Richard J. Boucherie, and M.J.A.L. van Kraaij, considers the runway selection problem that is typical for airports with a complex layout of runways. This chapter describes a stochastic dynamic programming (SDP) approach determining an optimal strategy for the monthly preference list selection problem under safety and efficiency restrictions and yearly noise load restrictions, as well as future and unpredictable weather conditions. As special MDP complications, a continuous state (noise volume) has to be discretized, and other states at sufficient distance are lumped to make the SDP numerically tractable.

In Chap. 12, by Mark Fackrell and Peter Taylor, both public and private goals are optimized, the latter indirectly. The objective is to balance the distribution of cars in a vertical car park by allocating arriving cars to levels in the best way. If no place is available, a car arrival is assumed to be lost. The randomness is inherent in the arrival process and the parking durations. This daily life problem implicitly concerns the problem of job allocation in an overflow system, a class of problems which are known to be unsolvable analytically in the uncontrolled case. An MDP heuristic rule is developed and extensive experiments show it to be superior.

Chapter 13, by Rene Haijema, Eligius M.T. Hendrix, and Jan van der Wal, studies another problem of daily life and both public and private concerns: dynamic control of traffic lights to minimize the mean waiting time of vehicles. The approach involves an approximate solution for a multidimensional MDP based on policy iteration in combination with decomposition of the state space into state spaces for different traffic streams. Numerical results illustrate that a single policy iteration step results in a strategy that greatly reduces average waiting time when compared to static control.

The final chapter of this transportation category, Chap. 14, by Pia L. Kempker, Nico M. van Dijk, Werner Scheinhardt, Hans van den Berg, and Johann Hurink, addresses overnight charging of electric vehicles taking into account the fluctuating energy demand and prices. A heuristic bidding strategy that is based on an analytical solution of the SDP for i.i.d. prices shows a substantial performance improvement compared to currently used standard demand side management strategies.

Part IV: Production

Control of production systems is a well-known application area that is known to be hampered by its computational complexity. This part contains three cases that illustrate the structure of approximate policies:

Chapter 15: Analysis of a Stochastic Lot Scheduling Problem with Strict DueDates
Chapter 16: Optimal Fishery Policies
Chapter 17: Near-Optimal Switching Strategies for a Tandem Queue

Chapter 15, by Nicky D. Van Foreest and Jacob Wijngaard, considers admission control and scheduling rules for a make-to-order stochastic lot scheduling problem with strict due dates. The CSLSP is a difficult scheduling problem for which MDPs seem to be one of the few approaches to analyze this problem. The MDP formulation further allows to set up simulations for large-scale systems.

In Chap. 16, by Eligius, M.T. Hendrix, Rene Haijema, and Diana van Dijk, a bi-level MDP for optimal fishing quota is studied. At the first level, an authority decides on the quota to be fished keeping in mind long-term revenues. At the second level, fishermen react on the quota set as well as on the current states of fish stock and fleet capacity by deciding on their investment and fishery effort. This chapter illustrates how an MDP with continuous state and action space can be solved by truncation and discretization of the state space and applying interpolation in the value iteration.

Chapter 17, by Daphne van Leeuwen and Rudesindo Núñez-Queija, is motivated by applications in logistics, road traffic, and production management. This chapter considers a tandem network, in which the waiting costs in the second queue are larger than those in the first queue. MDP is used to determine the near-optimal switching curve between serving and not serving at the first queue. that balances waiting costs at the queues. Discrete event simulation is used to show the appropriateness of the near-optimal strategies.

Part V: Communications

Communications has been an important application area for MDP with particular emphasis on call acceptance rules, channel selection, and transmission rates. This part illustrates some special cases for which a (near)-optimal strategy can be obtained:

Chapter 18: Wireless Channel Selection with Restless Bandits
Chapter 19: Flexible Staffing for Call Centers With Non-stationary Arrival Rates
Chapter 20: MDP for Query-Based Wireless Sensor Networks

Chapter 18, by Julia Kuhn and Yoni Nazarathy, considers wireless channel selection to maximize the long-run average throughput. The online control problem is modeled as restless multi-armed bandit (RMAB) problem in a POMDP framework. The chapter unifies several approaches and presents a nice development of the Whittle index.

Chapter 19, by Alex Roubos, Sandjai Bhulai, and Ger Koole, develops an MDP to obtain time-dependent staffing levels in a single-skill call center such that a service-level constraint is met in the presence of time-varying arrival rates. Through a numerical study based on real-life data, it is shown that the optimal policies provide a good balance between staffing costs and the penalty probability for not meeting the service level.

Chapter 20, by Mihaela Mitici, studies queries in a wireless sensor network, where queries might either be processed within the sensor network with possible delay or queries might be allocated to a database without delay but possibly containing outdated data. An optimal policy for query assignment is obtained from a continuous time MDP with drift. By an exponentially uniformized conversion (as extension of standard uniformization), it is transformed into a standard discrete-time MDP. By computation this leads to close-to-optimal simple policies.

Part VI: Financial Modeling

It is needless to say that financial modeling and stochastics are intrinsically related. Financial models represent a major field with time-series analysis for long-term financial and economic purposes as one well-known direction. Related directions concern stock, option, and utility theory. Early decision theory papers on portfolio management and investment modeling date back to the 1970s; see the edited book.[7] From a pure MDP perspective, the recently published book on Markov decision processes with special application in finance,[8] and the earlier papers by Jörn Sass and Manfred Schäl are recommended.

Chapter 21 by Jörn Sass and Manfred Schäl, gives an instructive review and follow-up on their earlier work to account for financial portfolios and derivatives under proportional transactional costs. In particular, a computational algorithm is developed for optimal pricing, and the optimal policy is shown to be a martingale that is of special interest in financial trading.

[7] Michael A. H. Dempster and Stanley R. Pliska, editors. *Mathematics of Derivative Securities*. Cambridge University Press, 1997.

[8] N. Bäuerle, U. Rieder. *Markov Decision Processes with Applications in Finance*. Springer, 2011.

Summarizing

These practical MDP applications have illustrated a variety of both standard and nonstandard aspects of MDP modeling and its practical use:

- A first and major step is a proper state definition containing sufficient information and details, which will frequently lead to multidimensional discrete or continuous states.
- The transition structure of the underlying process may involve time-dependent transition probabilities.
- The objective for optimization may be an average, discounted, or finite-time criterion.
- One-step rewards may be time dependent.
- The action set may be continuous or discrete.
- A simplified but computationally solvable situation can be an important first step in deriving a suitable policy that may subsequently be expanded to the solution of a more realistic case.
- Heuristic policies that may be implemented in practice can be developed from optimal policies.

We are confident that this handbook is appealing for a variety of readers with a background in, among others, operations research, mathematics, computer science, and industrial engineering:

1. A practitioner that would like to become acquainted with the possible value of MDP modeling and ways to use it
2. An academic or institutional researcher to become involved in an MDP modeling and development project and possibly expanding its frontiers
3. An instructor or student to be inspired by the instructive examples in this handbook to start using MDP for real-life problems

From each of these categories you are invited to step in and enjoy reading this hand book for further practical MDP applications.

P. Dorreboom 2009†: DP – a repetitive structure

Acknowledgments

We are most grateful to all authors for their positive reactions right from the initial invitations to contribute to this handbook: it is the quality of the chapters and the enthusiasm of the authors that will enable MDP to have its well-deserved impact on real-life applications.

We like to deeply express our gratitude to the former editor in chief and series editor: Fred Hillier. Had it not been for his stimulation from the very beginning in the first place and his assistance in its handling for approval, just before retirement, we would not have succeeded to complete this handbook.

Enschede, The Netherlands
July 2016

Richard J. Boucherie
Nico M. van Dijk

Contents

Part V Communications

List of Contributors

S. Bhulai
Faculty of Sciences, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands

H. Blok
Eindhoven University of Technology, Eindhoven, The Netherlands

Richard J. Boucherie
Stochastic Operations Research, University of Twente, Enschede, The Netherlands

Brian T. Denton
Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI, USA

Mark Fackrell
School of Mathematics and Statistics, University of Melbourne, VIC, Australia

Rene Haijema
Operations Research and Logistics group, Wageningen University, Wageningen, The Netherlands

Eligius M.T. Hendrix
Computer Architecture, Universidad de Málaga, Málaga, Spain

Johann Hurink
Department of Applied Mathematics, University of Twente, Enschede, The Netherlands

M.J. IJzerman
Department of Health Technology and Services Research, University of Twente, Enschede, The Netherlands

C.J. Jagtenberg
Stochastics, CWI, Amsterdam, The Netherlands

Pia L. Kempker
TNO, Cyber Security & Robustness TNO, The Hague, The Netherlands

Ger Koole
Faculty of Sciences, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands

Julia Kuhn
The University of Queensland, Brisbane, QLD, Australia

University of Amsterdam, Amsterdam, The Netherlands

T.R. Meerburg
Air Traffic Control The Netherlands, Schiphol, The Netherlands

Martijn Mes
Department of Industrial Engineering and Business Information Systems, University of Twente, Enschede, The Netherlands

András Mészáros
MTA-BME Information Systems Research Group, Budapest, Hungary

Mihaela Mitici
Faculty of Aerospace Engineering, Air Transport and Operations, Delft University of Technology, Delft, The Netherlands

Yoni Nazarathy
The University of Queensland, Brisbane, QLD, Australia

Rudesindo Núñez-Queija
CWI, Amsterdam, The Netherlands

J.W.M. Otten
Department of Stochastic Operations Research, University of Twente, Enschede, The Netherlands

Arturo Pérez Rivera
Department of Industrial Engineering and Business Information Systems, University of Twente, Enschede, The Netherlands

Martin L. Puterman
Sauder School of Business, University of British Columbia, Vancouver, BC, Canada V6T 1Z2

Alex Roubos
CCmath, Amsterdam, The Netherlands

Jörn Sass
Fachbereich Mathematik, TU Kaiserslautern, Kaiserslautern, Germany

Antoine Sauré
Telfer School of Management, University of Ottawa, Ottawa, ON, Canada K1N 6N5

Manfred Schäl
Institut für Angewandte Mathematik, Universität Bonn, Bonn, Germany

Werner Scheinhardt
Department of Applied Mathematics, University of Twente, Enschede, The Netherlands

S. Siesling
Department of Health Technology and Services Research, University of Twente, Enschede, The Netherlands

Department of Research, Comprehensive Cancer Organisation, Utrecht, The Netherlands

F.M. Spieksma
Leiden University, Leiden, The Netherlands

Lauren N. Steimle
Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI, USA

Peter Taylor
School of Mathematics and Statistics, University of Melbourne, VIC, Australia

Miklós Telek
Budapest University of Technology and Economics, Budapest, Hungary

Henk Tijms
Vrije Universiteit Amsterdam, Amsterdam, The Netherlands

J.B. Timmer
Department of Stochastic Operations Research, University of Twente, Enschede, The Netherlands

Nico M. van Dijk
Stochastic Operations Research, University of Twente, Enschede, The Netherlands

Diana van Dijk
Department of Environmental Social Sciences, Swiss Federal Institute of Aquatic Science and Technology (EAWAG), Dübendorf, Switzerland

Nicky D. van Foreest
Faculty of Economics and Business, University of Groningen, Groningen, The Netherlands

M.J.A.L. van Kraaij
Air Traffic Control, Utrecht, The Netherlands

Daphne van Leeuwen
CWI, Amsterdam, The Netherlands

Hans van den Berg
TNO, Cyber Security & Robustness TNO, The Hague, The Netherlands

Department of Applied Mathematics, University of Twente, Enschede, The Netherlands

R.D. van der Mei
Stochastics, CWI, Amsterdam, The Netherlands

Jan van der Wal
Faculty of Economics and Business, University of Amsterdam, Amsterdam, The Netherlands

Stochastic Operations Research group, University of Twente, Enschede, The Netherlands

I.M.H. Vliegen
Department of Industrial Engineering and Business Information Systems, University of Twente, Enschede, The Netherlands

A. Witteveen
Department of Health Technology and Services Research, University of Twente, Enschede, The Netherlands

Jacob Wijngaard
Faculty of Economics and Business, University of Groningen, Groningen, The Netherlands

Part I
General Theory

Chapter 1
One-Step Improvement Ideas and Computational Aspects

Henk Tijms

Abstract In this contribution we give a down-to-earth discussion on basic ideas for solving practical Markov decision problems. The emphasis is on the concept of the policy-improvement step for average cost optimization. This concept provides a flexible method of improving a given policy. By appropriately designing the policy-improvement step in specific applications, tailor-made algorithms may be developed to generate the best control rule within a class of control rules characterized by a few parameters. Also, in decision problems with an intractable multi-dimensional state space, decomposition and a once-only application of the policy-improvement step may lead to a good heuristic rule. These useful features of the policy-improvement concept will be illustrated with a queueing control problem with variable service rate and with the dynamic routing of arrivals to parallel queues. In the final section, we deal with the concept of the one-stage-look-ahead rule in optimal stopping and give several applications.

1.1 Introduction

Let's begin with two stochastic optimization problems of general interest.

- The first problem is how to assign randomly arriving messages or jobs to one of several groups of servers with different service rates. What assignment rule minimizes the long-run average waiting time per message or job? This Markov decision problem occurs in a variety of practical areas such as telecommunication networks, call centers, flexible manufacturing, and health care.

H. Tijms (✉)
Vrije Universiteit Amsterdam, Amsterdam, The Netherlands
e-mail: tijms@quicknet.nl

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_1

- The second problem is the classical Chow-Robbins stopping problem. A fair coin is repeatedly tossed with no bound on the number of tosses. The tossing process can be stopped at any moment and the payoff is the proportion of heads obtained at the moment of stopping. The goal is to find a stopping rule maximizing the expected value of the payoff.

What do have these two problems in common? The answer is that the optimal decision rule is difficult to obtain in both problems, whereas in each of the two problems a suboptimal decision rule is easily found by using the same approach of looking only one step forward in the decision process to see whether an improvement is possible.

This paper gives a survey of the basic concepts underlying computational approaches for Markov decision problems and optimal stopping problems. We first discuss computational methods for the Markov decision model with the long-run average cost per unit time as optimality criterion. The basic concept of relative values is introduced and it is shown how this concept can be used to improve a given policy. The idea of the policy-improvement step is flexible and powerful. It enables us to give bounds on the costs of the policies generated in the value-iteration algorithm and to design a tailor-made policy-iteration algorithm in specific applications of the Markov decision model. In multi-dimensional Markov decision applications it is usually not practically feasible to compute an optimal policy, but a clever once-only application of the policy-improvement step usually leads to a good heuristic rule. An example will be given to illustrate this powerful method. To conclude this paper, we discuss optimal stopping problems and show the usefulness of the concept of the one-stage-look-ahead rule.

1.2 The Average-Cost Markov Decision Model

In the classical book of Howard [9] the average cost Markov decision model was introduced. A dynamic system is reviewed at equidistant points of time $t = 0, 1, \ldots$. At each review the system is classified into one of a possible number of states and subsequently a decision has to be made. The set of possible states is denoted by I. For each state $i \in I$, a set $A(i)$ of decisions or actions is given. The state space I and the action sets $A(i)$ are assumed to be *finite*. The economic consequences of the decisions taken at the review times (decision epochs) are reflected in costs. This controlled dynamic system is called a *discrete-time Markov model* when the following Markovian property is satisfied. If at a decision epoch the action a is chosen in state i, then regardless of the past history of the system, the following happens:

(a) an immediate cost $c_i(a)$ is incurred,
(b) at the next decision epoch the system will be in state j with probability $p_{ij}(a)$, where $\sum_{j \in I} p_{ij}(a) = 1$ for $i \in I$.

The one-step costs $c_i(a)$ and the one-step transition probabilities $p_{ij}(a)$ are assumed to be time homogeneous. In specific problems the 'immediate' costs $c_i(a)$ will often represent the expected cost incurred until the next decision epoch when action a is

chosen in state i. The choice of the state space and of the action sets often depends on the cost structure of the specific problem considered.

A rule or policy for controlling the system is a prescription for taking actions at each decision epoch. In principle a control rule may be quite complicated in the sense that the prescribed actions may depend on the whole history of the system. An important class of policies is the subclass of stationary policies. A *stationary policy* R is a policy that assigns to each state i a fixed action $a = R_i$ and always uses this action whenever the system is in state i. In view of the Markov assumption made and the fact that the planning horizon is infinitely long, it will be intuitively clear that it is sufficient to consider only the class of stationary policies. However, other policies are conceivable: policies whose actions depend on the past states or policies whose actions are determined by a random mechanism. This issue touches a fundamental question in Markov decision theory: does there exist an optimal policy among the class of all conceivable policies and, if an optimal policy exists, is such a policy a stationary policy? The answer to these questions is in the affirmative for the finite-state and finite-action Markov decision model with the average cost criterion. However, a mathematical proof requires rather deep arguments. The interested reader is referred to Derman [4] and Puterman [15].

Let X_n be the state of the system at time n just prior to a decision. Under a stationary policy R, the process $\{X_n\}$ is a discrete-time Markov chain with one-step transition probabilities

$$P\{X_{n+1} = j \mid X_n = i\} = p_{ij}(R_i).$$

This Markov chain incurs a cost $c_i(R_i)$ each time the system visits state i. Thus we can invoke results from Markov chain theory to specify the long-run average cost per time unit under a given stationary policy. Unless stated otherwise, the following assumption is made throughout this chapter.

Unichain assumption. *For each stationary policy R, the associated Markov chain $\{X_n\}$ has a single recurrent class of states.*

In most practical applications the Markov chain $\{X_n\}$ is unichain for each stationary policy. The unichain assumption allows for transient states. Under this assumption, the Markov chain $\{X_n\}$ has a unique equilibrium distribution for each stationary policy. The equilibrium probabilities under stationary policy R are denoted by $\pi_j(R)$ for $j \in I$ and are the unique solution to the equilibrium equations

$$\pi_j(R) = \sum_{k \in I} \pi_k(R) p_{kj}(R_k) \quad \text{for } j \in I$$

together with the normalization equation $\sum_{j \in I} \pi_j(R) = 1$. By a well-known ergodic theorem from Markov chain theory, the long-run average cost per unit time under a stationary policy R is given by

$$g(R) = \sum_{j \in I} c_j(R_j) \pi_j(R),$$

independently of the starting state of the Markov chain. A stationary policy R^* is said to be *average cost optimal* if $g(R^*) \leq g(R)$ for each stationary policy R. As pointed out before, policy R^* is not only optimal among the class of stationary policies but it is also optimal among the class of all conceivable policies.

1.2.1 The Concept of Relative Values

There is an alternative approach to compute the average cost $g(R)$ of a given stationary policy R. This approach yields a so-called relative-value function which is the basis for an improvement of policy R.

Theorem 1. *Let R be a given stationary policy and r be a recurrent state of the Markov chain $\{X_n\}$ associated with policy R. Define the relative-value function $w_i(R)$ by*

$$w_i(R) = K_i(R) - g(R)T_i(R) \quad \text{for } i \in I,$$

where $T_i(R)$ is the expected time until the first transition into state r beyond time 0 when the initial state is i and policy R is used, and $K_i(R)$ is the expected costs incurred during this time with the convention that $K_i(R)$ includes the cost incurred in the initial state i at time 0 but excludes the cost incurred at the epoch of the first transition into state r. Then,

$$w_r(R) = 0.$$

The average cost $g(R)$ and the relative-value function $w_i(R)$ satisfy the linear equations

$$v_i = c_i(R_i) - g + \sum_{j \in I} p_{ij}(R_i)v_j \quad \text{for } i \in I.$$

The solution to these so-called value-determination is uniquely determined up to an additive constant for the function v_i, $i \in I$.

Proof. A Markov chain is a regenerative stochastic process and any recurrent state is a regeneration state. Letting a cycle be the time elapsed between two consecutive visits to the regeneration state r, it follows from the theory of renewal-reward processes that the average cost per time unit equals the expected costs incurred in one cycle divided by the expected length of one cycle and so $g(R) = \frac{K_r(R)}{T_r(R)}$, implying that $w_r(R) = 0$. By a conditioning argument,

$$T_i(R) = 1 + \sum_{j \in I, j \neq r} T_j(R)p_{ij}(R_i) \quad \text{for } i \in I,$$

$$K_i(R) = c_i(R_i) + \sum_{j \in I, j \neq r} K_j(R)p_{ij}(R_i) \quad \text{for } i \in I.$$

Multiplying both sides of the first equation with $g(R)$ and subtracting the resulting two equations, we get $w_i(R) = c_i(R_i) - g(R) + \sum_{j \in I, j \neq r} w_j(R)p_{ij}(R_i)$ for $i \in I$.

Noting that $w_r(R) = 0$, we have verified that $g(R)$ and $w_i(R)$, $i \in I$ satisfy the value-determination equations. Let g and v_i, $i \in I$ by any solution to these equations. Multiplying both sides of the equations by $\pi_i(R)$, summing over i and using the equilibrium equations for the $\pi_i(R)$, it follows after an interchange of the order of summation that $g = \sum_{i \in I} c_i(R_i)\pi_i(R)$, showing that $g = g(R)$. To prove that the relative-value function is uniquely determined up to an additive constant, let $\{g, v_i\}$ and $\{g, w_i\}$ be any two solutions to the value-determination equations. It is convenient to use matrix notation with $\mathbf{v} = (v_i)$, $\mathbf{w} = (w_i)$ and $\mathbf{P} = (p_{ij}(R_i))$. Then $\mathbf{v} - \mathbf{w} = \mathbf{P}(\mathbf{v} - \mathbf{w})$. Iterating this equation, we get $\mathbf{v} - \mathbf{w} = \mathbf{P}^n(\mathbf{v} - \mathbf{w})$ for all $n \geq 1$. This gives $\mathbf{v} - \mathbf{w} = \mathbf{Q}^n(\mathbf{v} - \mathbf{w})$ for all $n \geq 1$, where $\mathbf{Q}^n = (1/n)\sum_{k=1}^n \mathbf{P}^k$. It is well-known from Markov chain theory that the (i, j)th element of $\mathbf{Q}^n$ converges to $\pi_j(R)$ as $n \to \infty$ for all $i, j \in I$. This shows that $v_i - w_i = \sum_{j \in I}(v_j - w_j)\pi_j(R)$ for all $i \in I$, proving that the relative-value function is uniquely determined up to an additive constant.

Why the name relative-value function? To answer this question, we first state the following result.

Theorem 2. *For a fixed stationary policy R, define $V_n(i,R)$ as the total expected cost incurred over the first n decision epochs when the starting state is i. Then, under the assumption that the Markov chain $\{X_n\}$ associated with policy R is aperiodic, there exists a finite function $v_i(R)$ such that*

$$\lim_{n \to \infty} V_n(i,R) - ng(R) = v_i(R) \quad \text{for } i \in I.$$

Moreover, $g(R)$ and the $v_i(R)$ satisfy the value-determination equations from Theorem 1.

Proof. A sketch of the proof is as follows. Denoting by $p_{ij}^{(k)}(R)$ the k-step transition probabilities of the Markov chain $\{X_n\}$ associated with policy R, we have $V_n(i,R) = \sum_{k=0}^{n-1}\sum_{j \in I} p_{ij}^{(k)}(R)c_j(R_j)$, where $p_{ij}^{(0)}$ is 1 for $j = i$ and is 0 otherwise. Together with the representation $g(R) = \sum_{j \in I} c_j(R_j)\pi_j(R)$, this leads after an interchange of the order of summation to

$$V_n(i,R) - ng(R) = \sum_{j \in I} c_j(R_j) \sum_{k=0}^{n-1} \left[p_{ij}^{(k)}(R) - \pi_j(R)\right].$$

We now invoke the assumption that the Markov chain $\{X_n\}$ associated with policy R is aperiodic. Then, the k-step transition probability $p_{ij}^{(k)}(R)$ converges exponentially fast to $\pi_j(R)$ as $k \to \infty$ for all $i, j \in I$, see e.g. Theorem 3.5.12 in Tijms [19]. That is, there are constants $\alpha > 0$ and $0 < \beta < 1$ such that $|p_{ij}^{(k)}(R) - \pi_j(R)| \leq \alpha\beta^k$ for all $k \geq 1$ and $i, j \in I$. This gives that the series $\sum_{k=0}^{\infty}\left[p_{ij}^{(k)}(R) - \pi_j(R)\right]$ is absolutely convergent, implying that $\lim_{n \to \infty}\sum_{k=0}^{n-1}\left[p_{ij}^{(k)}(R) - \pi_j(R)\right]$ exists and is finite for all $i \in I$, proving that $V_n(i,R) - ng(R)$ has a finite limit $v_i(R)$ as $n \to \infty$. To verify that $v_i(R)$, $i \in I$ is a relative-value function, use the recursive equation

$$V_n(i,R) = c_i(R_i) + \sum_{j \in I} p_{ij}(R_i) V_{n-1}(j,R),$$

subtract $ng(R)$ from its both sides and let $n \to \infty$ to obtain that the $v_i(R)$ satisfy the value-determination equations for policy R.

We can now explain the term relative-value function. Theorem 2 implies that

$$\lim_{n \to \infty} V_n(i,R) - V_n(j,R) = v_i(R) - v_j(R) \quad \text{for all } i,j \in I.$$

In other words, using the fact that the relative-value function is uniquely determined up to an additive constant, we have for any relative-value function v_i, $i \in I$ for policy R that $v_i - v_j$ represents the difference in the total expected costs over the infinite planning period $t = 1,2,\ldots$ when the starting state is i rather than j provided that the Markov chain $\{X_n\}$ associated with policy R is aperiodic.

1.2.2 The Policy-Improvement Step

In this section we come to the most important solution tool in Markovian control. This tool is the policy-improvement step and provides a flexible method to improve a given stationary policy R in order to obtain a stationary policy with a lower average cost per unit time. We first give a heuristic motivation of the policy-improvement step and next prove that it indeed leads to a better policy. The heuristic idea to improve a given stationary policy R is as follows. Suppose that you ask yourselves the question "how does change the total expected cost over the first n decision epochs when deviating from policy R by taking some other decision a in state i at the first decision epoch and next using policy R over the remaining decision epochs?" The change in the total expected costs over the first n decision epochs is given by

$$c_i(a) + \sum_{j \in I} p_{ij}(a) V_{n-1}(j,R) - V_n(i,R).$$

For the moment, assume that the Markov chain $\{X_n\}$ associated with policy R is aperiodic. Then, by the results in Remark 1 in the previous section, we have that $V_n(i,R) \approx ng(R) + v_i(R)$ for n large enough. Inserting this into the above expression, we get that the change in the expected costs is approximately given by $c_i(a) + \sum_{j \in I} p_{ij}(a) v_j(R) - g(R) - v_i(R)$. This suggests to look for an action a in state i so that the so-called *policy-improvement inequality*

$$c_i(a) - g(R) + \sum_{j \in I} p_{ij}(a) v_j(R) \leq v_i(R)$$

is satisfied. This heuristic discussion motivates our main theorem.

Theorem 3 (Improvement Theorem). *Let g and v_i, $i \in I$, be given numbers. Suppose that the stationary policy $\overline{R}$ has the property*

$$c_i(\overline{R}_i) - g + \sum_{j \in I} p_{ij}(\overline{R}_i) v_j \leq v_i \quad \textit{for all } i \in I.$$

Then the long-run average cost of policy $\overline{R}$ satisfies

$$g(\overline{R}) \leq g$$

with strict inequality if the strict inequality sign holds in the policy-improvement inequality for some state i that is recurrent under policy $\overline{R}$.

Proof. Since the Markov chain $\{X_n\}$ associated with policy $\overline{R}$ is unichain, this Markov chain has a unique equilibrium distribution $\{\pi_i(\overline{R}), i \in I\}$. The equilibrium probability $\pi_i(\overline{R})$ is positive only if state i is recurrent under policy $\overline{R}$. Multiplying both sides of the policy-improvement inequality by $\pi_i(\overline{R})$, summing over i and noting that $\sum_{i \in I} \pi_i(\overline{R}) = 1$, we get

$$\sum_{i \in I} \pi_i(\overline{R}) c_i(\overline{R}_i) - g + \sum_{i \in I} \pi_i(\overline{R}) \sum_{j \in I} p_{ij}(\overline{R}_i) v_j \leq \sum_{i \in I} \pi_i(\overline{R}) v_i,$$

where the strict inequality sign holds if there is strict inequality in the policy-improvement inequality for some state i with $\pi_i(\overline{R}) > 0$. Interchanging the order of summation in the double sum and using the equilibrium equations $\pi_j(\overline{R}) = \sum_{i \in I} \pi_i(\overline{R}) p_{ij}(\overline{R}_i)$ together with $g(\overline{R}) = \sum_{i \in I} \pi_i(\overline{R}) c_i(\overline{R}_i)$, we find

$$g(\overline{R}) - g + \sum_{j \in I} \pi_j(\overline{R}) v_j \leq \sum_{i \in I} \pi_i(\overline{R}) v_i,$$

where the strict inequality sign holds if there is strict inequality in the policy-improvement inequality for some state i which is recurrent for policy $\overline{R}$. This completes the proof.

Remark 1. An examination of the proof shows that Theorem 3 remains valid when all inequality signs are reversed. As a consequence, a stationary policy R with average cost $g(R)$ and relative values $v_i(R)$, $i \in I$ is average cost optimal if

$$c_i(a) - g(R) + \sum_{j \in I} p_{ij}(a) v_j(R) \leq v_i(R) \quad \text{for all } i \in I \text{ and } a \in A(i).$$

Then $g(R)$ and $v_i(R)$, $i \in I$ satisfy the so-called *average cost optimality equation*

$$v_i = \min_{a \in A(i)} \left\{ c_i(a) - g + \sum_{j \in I} p_{ij}(a) v_j \right\} \quad \text{for } i \in I.$$

An interesting interpretation can be attached to the policy-improvement inequality. Suppose that a control cost of $c_i(a) - g$ is incurred when action $a = \overline{R}_i$ is chosen in state i, while a terminal cost of v_j is incurred when the control of the system is stopped and the system is left behind in state j. Then the policy-improvement inequality states that controlling the system for one step according to rule $\overline{R}$ and stopping next is preferable to stopping directly.

The policy-improvement step is an extremely important concept in Markovian control. This concept allows for a flexible application. It can be used to derive the so-called Odoni bounds for the value-iteration algorithm. This is a powerful algorithm that can be used to any Markovian control application and this algorithm takes its practical usefulness from the Odoni bounds. These bounds and the value-iteration algorithm will be discussed in the next section. In applications, one wants sometimes to confine to control rules that have a specific structure and are described by a few parameters. For example, (s,S) ordering policies in controlled inventory systems or policies with one or two switch-over levels to control the queue size in queueing systems for which the service rate or arrival rate can be controlled. Designing appropriately the policy-improvement step, a tailor-made algorithm can be developed to generate a sequence of improved policies having the desired structure. An application of this approach will be given in Sect. 1.4. The policy-improvement procedure has the remarkable feature that it achieves the largest cost improvements in the first few iterations when it is repeatedly applied to generate a sequence of policies. This finding underlies a heuristic approach for Markov decision problems with a multi-dimensional state space. In such decision problems it is usually not feasible to solve the value-determination equations. However, a policy-improvement step offers in general no computational difficulties. This suggests a heuristic approach that determines first a good estimate for the relative values of an appropriately chosen policy and next applies only *one* policy-improvement step. This heuristic approach often results in a very good suboptimal rule. An application of this approach will be given in Sect. 1.5.

Remark 2. The applications that will be discussed in the Sects. 1.4 and 1.5 deal with the *semi-Markov decision model*. This model differs from the discrete-time Markov decision model only in the feature that the decision epochs occur randomly in time:

$\tau_i(a)$ = the expected time until the next decision epoch if action a is chosen in the current state i.

It is assumed that $\tau_i(a) > 0$ for all i,a. As before, the random variable X_n is defined as the state of the system just prior to the nth decision process. The embedded process $\{X_n\}$ is a discrete-time Markov chain under each stationary policy R. Assuming that this Markov chain is unichain, the long-run average cost per unit time under policy R is now given by

$$g(R) = \sum_{j\in I} c_j(R_j)\pi_j(R) \Big/ \sum_{j\in I} \tau_j(R_j)\pi_j(R)$$

with $\{\pi_j(R)\}$ again denoting the equilibrium distribution of the embedded Markov chain $\{X_n\}$. An examination of the proof of Theorem 1 reveals that this theorem remains valid for the semi-Markov decision model provided that the value-determinations equations are adjusted as

$$v_i = c_i(R_i) - g\tau_i(R_i) + \sum_{j\in I} p_{ij}(R_i)v_j \quad \text{for } i \in I.$$

Also, Theorem 3 remains valid provided that the policy-improvement inequality is adjusted as

$$c_i(\overline{R}_i) - g\tau_i(\overline{R}_i) + \sum_{j \in I} p_{ij}(\overline{R}_i)v_j \leq v_i \quad \text{for all } i \in I.$$

1.2.3 The Odoni Bounds for Value Iteration

The value-iteration algorithm computes recursively a sequence of value functions approximating the minimum average cost per time unit. The value functions can be used to give lower and upper bounds on the minimum average cost and the average cost of the generated policies. These bounds are the so-called Odoni bounds named after Odoni [13], see also Hastings [7]. The bounds converge to the minimum average cost under an aperiodicity condition. The value-iteration algorithm endowed with these lower and upper bounds is in general the best computational method for solving large-scale Markov decision problems. Using Theorem 3 in Sect. 1.2, we will give a simple derivation of these bounds. Before doing this, let us first formulate the value-iteration algorithm. The value-iteration algorithm computes recursively for $n = 1, 2, \ldots$ the value function $V_n(i)$ from

$$V_n(i) = \min_{a \in A(i)} \left\{ c_i(a) + \sum_{j \in I} p_{ij}(a) V_{n-1}(j) \right\} \quad \text{for } i \in I,$$

starting with an arbitrarily chosen function $V_0(i)$, $i \in I$. The quantity $V_n(i)$ can be interpreted as the minimum total expected costs with n periods left to the time horizon when the current state is i and a terminal cost of $V_0(j)$ is incurred when the system ends up at state j; see e.g. [3] for a proof. Intuitively, one might expect that the one-step difference $V_n(i) - V_{n-1}(i)$ will come very close to the minimum average cost per time unit and that the stationary policy whose actions minimize the right side of the equation for $V_n(i)$ for all i will be very close in cost to the minimum average cost when n is large enough. The recursion equation for $V_n(i)$ suggests to investigate the operator T that adds to each function $\mathbf{v} = (v_i,\ i \in I)$ a function $T\mathbf{v}$ whose ith component $(T\mathbf{v})_i$ is defined by

$$(T\mathbf{v})_i = \min_{a \in A(i)} \left\{ c_i(a) + \sum_{j \in I} p_{ij}(a) v_j \right\} \quad \text{for } i \in I.$$

Note that $(T\mathbf{v})_i = V_n(i)$ if $v_i = V_{n-1}(i)$, $i \in I$. The following theorem plays a key role in the value-iteration algorithm.

Theorem 4. *Let $\boldsymbol{v} = (v_i, i \in I)$ be a given function. Define the stationary policy $R(\boldsymbol{v})$ as a policy which adds to each state $i \in I$ an action $a = R_i(\boldsymbol{v})$ that minimizes the right-hand side of the equation for $(T\boldsymbol{v})_i$. Then,*

$$\min_{i \in I} \{(Tv)_i - v_i\} \leq g^* \leq g(R(\boldsymbol{v})) \leq \max_{i \in I} \{(Tv)_i - v_i\}$$

where g^* is the minimum long-run average cost per unit time.

Proof. To verify the bounds on g^*, take any stationary policy R. By the definition of $(T\mathbf{v})_i$, we have for any state $i \in I$ that

$$(T\mathbf{v})_i \le c_i(a) + \sum_{j \in I} p_{ij}(a) v_j \quad \text{for all } a \in A(i),$$

where the equality sign holds for $a = R_i(\mathbf{v})$. Choosing $a = R_i$ gives

$$(Tv)_i \le c_i(R_i) + \sum_{i \in I} p_{ij}(R_i) v_j \quad \text{for } i \in I.$$

Let $m = \min_{i \in I}\{(Tv)_i - v_i\}$. Noting that $m \le (Tv)_i - v_i$ for all i and using the above inequality, we get $m + v_i \le c_i(R_i) + \sum_{j \in I} p_{ij}(R_i) v_j$ for all $i \in I$, and so

$$c_i(R_i) - m + \sum_{j \in I} p_{ij}(R_i) v_j \ge v_i \quad \text{for } i \in I.$$

An application of Theorem 3 now gives that $g(R) \ge m$. This inequality holds for each policy R and so $g^* = \min_R g(R) \ge m$. The derivation of the upper bound for $g(R(\mathbf{v}))$ is very similar. By the definition of policy $R(\mathbf{v})$,

$$(T\mathbf{v})_i = c_i(R_i(\mathbf{v})) + \sum_{j \in I} p_{ij}(R_i(\mathbf{v})) v_j \quad \text{for } i \in I.$$

Let $M = \max_{i \in I}\{(T\mathbf{v})_i - v_i\}$. Since $M \ge (T\mathbf{v})_i - v_i$ for all $i \in I$, we get

$$c_i(R_i(\mathbf{v})) - M + \sum_{j \in I} p_{ij}(R_i(\mathbf{v})) v_j \le v_i \quad \text{for } i \in I.$$

Hence, by Theorem 3, $g(R(\mathbf{v})) \le M$ for all $i \in I$. This completes the proof.

How do the bounds in Theorem 4 work out for the value-iteration algorithm? Let $R^{(n)}$ be any stationary policy such that the action $a = R_i^{(n)}$ minimizes the right-hand side of the recursive equation for $V_n(i)$ for all $i \in I$. Define the Odoni bounds

$$m_n = \min_{i \in I}\{V_n(i) - V_{n-1}(i)\} \text{ and } M_n = \max_{i \in I}\{V_n(i) - V_{n-1}(i)\}.$$

Then, $m_n \le g^* \le g(R^{(n)}) \le M_n$. It is no restriction to assume that $c_i(a) > 0$ for all i, a; otherwise, add a same sufficiently large constant c to each $c_i(a)$. We then have $m_n > 0$ so that the bounds imply that

$$0 \le \frac{g(R^{(n)}) - g^*}{g^*} \le \frac{M_n - m_n}{m_n}.$$

The value-iteration algorithm is typically stopped as soon as $0 \le M_n - m_n \le \varepsilon m_n$, where $\varepsilon > 0$ is a small prespecified number, e.g. $\varepsilon = 10^{-3}$. The remaining question is whether the lower and upper bounds m_n and M_n converge to the same limit so

that the algorithm will be stopped after finitely many iterations. The answer to this question is only in the affirmative if a certain *aperiodicity* condition is satisfied for the underlying Markov chains. A sufficient condition is that the Markov chain $\{X_n\}$ is aperiodic for each average cost optimal stationary policy. Then the monotone sequences $\{m_n\}$ and $\{M_n\}$ have the same limit g^* and the convergence to this limit is exponentially fast, see Schweitzer and Federgruen [18], Van der Wal [18] and Tijms [19].

The aperiodicity requirement is no problem for the value-iteration method. The periodicity issue can be circumvented by a classical *perturbation* of the one-step transition probabilities of a Markov chain. If the one-step transition probabilities p_{ij} of a Markov chain are perturbed as $\overline{p}_{ij} = \tau p_{ij}$ for $j \neq i$ and $\overline{p}_{ii} = \tau p_{ii} + 1 - \tau$ for some constant τ with $0 < \tau < 1$, the perturbed Markov chain with one-step transition probabilities $\overline{p}_{ij}$ is aperiodic and has the same equilibrium probabilities as the original Markov chain. Thus a Markov decision model involving periodicity may be perturbed as follows. Choose some constant τ with $0 < \tau < 1$ and let the state space, the action sets, and the one-step costs unchanged. For any $i \in I$ and $a \in A(i)$, the one-step transition probabilities of the perturbed Markov decision model are defined by

$$p_{ii}(a) = \tau p_{ij}(a) + 1 - \tau \text{ and } p_{ij}(a) = \tau p_{ij}(a) \text{ for } j \neq i.$$

For each stationary policy, the associated Markov chain in the perturbed model is aperiodic, while the average cost per unit time in the perturbed model is the same as that in the original model. In specific problems involving periodicity the 'optimal' value of τ is usually not clear beforehand; empirical investigations indicate that $\tau = \frac{1}{2}$ is usually a satisfactory choice. It is noted that this transformation technique can be generalized to transform any semi-Markov decision model into an equivalent discrete-time Markov decision model so that the value-iteration method can also be applied to the semi-Markov decision model, see Tijms [19] for details.

Finally, it is pointed out that the unichain assumption from Sect. 1.1 can be weakened. In practical applications of the average cost Markov decision model, there may be nonoptimal policies for which the associated Markov chains have multiple recurrent classes, but the unichain property typically holds for the average cost optimal policies. For the value-iteration method, it suffices to require that the Markov chains associated with the average cost optimal policies are unichain (and aperiodic), see Tijms [19] for further details.

1.3 Tailor-Made Policy-Iteration Algorithm

Before formulating the queueing control problem, we discuss the general idea of an embedding technique that will used in this control problem. By exploiting the structure of the specific problem, the computational work in solving the

value-determination equations can be considerably reduced. This embedding technique will be discussed in the context of the semi-Markov decision model in which the decision epochs occur randomly in time and it suffices to know $\tau_i(a)$ being the expected time until the next decision epoch when action a is chosen in state i at the current decision epoch. For a fixed stationary policy R, let E be an appropriately chosen subset of the state space I such that under policy R the set E of states can be reached from each initial state $i \in I$. Define now for each $i \in I$ and $j \in E$,

$p_{ij}^E(R)$ = the probability that the first transition to a state in E is to state j when the initial state is i and policy R is used,

$\tau_i^E(R)$ = the time until the first transition to a state in E when the initial state is i and policy R is used,

$c_i^E(R)$ = the expected cost incurred until the first transition to a state in E when the initial state is i and policy R is used.

Theorem 5. *Let $r \in E$ be a state that is recurrent under policy R. Then the system of linear equations,*

$$v_i = c_i^E(R) - g\tau_i^E(R) + \sum_{j \in E} p_{ij}^E(R) v_j \quad \text{for } i \in E$$

together with the normalization $v_r = 0$ has a unique solution. Denote this solution by g^E and v_i^E for $i \in E$ and augment this solution by

$$v_i^E = c_i^E(R) - g\tau_i^E(R) + \sum_{j \in E} v_j p_{ij}^E(R) \quad \text{for } i \notin E.$$

Then $g^E = g(R)$. Moreover, g^E and the v_i^E, $i \in I$ satisfy the original value-determination equations $v_i = c_i(R_i) - g\tau_i(R_i) + \sum_{j \in I} p_{ij}(R_i) v_j$, $i \in I$.

Proof. The first assertion is a consequence of the version of Theorem 1 for the semi-Markov model, as follows by applying this theorem to the Markov process embedded on the set E. To prove the other assertions, let the function $w_i(R) = K_i(R) - g(R)T_i(R)$, $i \in I$ be defined as in Theorem 1. By the uniqueness of the solution g^E and the v_i^E and the fact that $g(R)$ and the $w_i(R)$ satisfy the value-determination equations for policy R (Theorem 1), it suffices to verify that

$$w_i(R) = c_i^E(R) - g(R)\tau_i^E(R) + \sum_{j \in E} w_j(R) p_{ij}^E(R) \quad \text{for all } i \in I.$$

This is easily shown. Using the definitions of $T_i(R)$ and $K_i(R)$, we have

$$T_i(R) = \tau_i^E(R) + \sum_{j \in E,\, j \neq r} p_{ij}^E(R) T_j(R) \quad \text{for } i \in I,$$

$$K_i(R) = c_i^E(R) + \sum_{j \in E,\, j \neq r} p_{ij}^E(R) K_j(R) \quad \text{for } i \in I.$$

Subtracting $g(R)$ times the equation for $T_i(R)$ from the equation for $K_i(R)$ and noting that $w_r(R) = 0$, we get the equation to be verified for the $w_i(R)$.

1.3.1 A Queueing Control Problem with a Variable Service Rate

Messages arrive at one of the outgoing communication lines in a message switching center according to a Poisson process with rate λ. The message length is exponentially distributed with mean $1/\mu$. An arriving message finding the communication line idle, is provided with service immediately; otherwise, the message is stored in a buffer until access to the line can be given. The communication line is only able to transmit one message at a time, but has available two possible transmission rates σ_1 and σ_2 with $0 \leq \sigma_1 < \sigma_2$. It is assumed that the faster transmission rate σ_2 is larger than the offered traffic λ/μ. At any time the line may switch from one transmission rate to the other. A fixed cost of $K \geq 0$ is incurred when switching from rate σ_1 to the faster rate σ_2. An operating cost at rate $r_i > 0$ is incurred when the line is transmitting a message using rate σ_i, while an operating cost at rate $r_0 \geq 0$ is incurred when the line is idle. For each message a holding cost of $h > 0$ is incurred for each unit of time the message is in the system until its transmission is completed. An easily implementable control is the (i_1, i_2) rule with $0 \leq i_2 < i_1$. Under this rule the communication line switches from transmission rate σ_1 to σ_2 when the number of messages in the system increases to the level i_1 at an arrival epoch, and switches from rate σ_2 to rate σ_1 when the number of messages in the system decreases to the level i_2 at a transmission completion epoch. The goal is to find the best rule within the class of the (i_1, i_2) rules, where the criterion is the long-run average cost per unit time.

The problem of controlling the transmission rate as a function of the number of messages in the system is a semi-Markov decision problem in which the decision epochs are given by the arrival epochs and the transmission completion epochs. The message lengths and the interarrival times of messages are both exponentially distributed and so, by the memoryless property of the exponential distribution, the state of the system can be described by $s = (i,k)$, where i denotes the number of messages in the system and $k = 1$ or 2 depending on whether the prevailing transmission rate is σ_1 or σ_2. The number of possible states is unbounded. However, this will offer no computational problems when using the embedding technique. To develop a tailor-made policy-iteration algorithm operating on the class of (i_1, i_2) rules, we first show how embedding leads to a very simple calculation of the long-run average cost and the relative values associated with an (i_1, i_2) rule. For a given rule R of the (i_1, i_2)-type, the embedded system of linear equations in Theorem 5 becomes quite simple when choosing as embedded set

$$E = \{(i_1, 1), (i_2, 2)\}.$$

The embedded Markov chain on E visits alternately the states $(i_1, 1)$ and $(i_2, 2)$. The transition probabilities $p^E_{st}(R)$ are given by

$$p^E_{(i,1),(i_1,1)}(R) = 1 \text{ for } 0 \leq i < i_1, \; p^E_{(i,1),(i_2,2)}(R) = 1 \text{ for } i \geq i_1,$$
$$p^E_{(i,2),(i_1,1)}(R) = 1 \text{ for } 0 \leq i \leq i_2, \; p^E_{(i,2),(i_2,2)}(R) = 1 \text{ for } i > i_2.$$

Next we specify the quantities $\tau_s^E(R)$ and $c_s^E(R)$. Obviously,

$$\tau_{(i,1)}^E(R) = \tau_{(i,2)}^E(R) \text{ for } i > i_1,\ c_{(i,1)}^E(R) = K + c_{(i,2)}^E(R) \text{ for } i \geq i_1.$$

Also, $\tau_{(i,2)}^E(R) = \tau_{(i,1)}^E(R)$ and $c_{(i,2)}^E(R) = c_{(i,1)}^E(R)$ for $0 \leq i \leq i_2$. The other $\tau_s^E(R)$ and $c_s^E(R)$ are specified in the following two lemmas.

Lemma 6. For $i > i_2$,

$$\tau_{(i,2)}^E(R) = \frac{i-i_2}{\sigma_2\mu - \lambda},\ c_{(i,2)}^E(R) = \frac{i-i_2}{\sigma_2\mu-\lambda}\Big[\frac{1}{2}h(i-i_2+1) + \frac{h\lambda}{\sigma_2\mu-\lambda} + r_2\Big].$$

Proof. The formulas can be directly obtained from a basic result in queueing theory. Take the $M/M/1$ queue with arrival rate λ and service rate η such that $\lambda/\eta < 1$, and assume that a holding cost at rate hi is incurred whenever there are i customers in the system. Then the time until the system becomes empty when starting with i customers in the system has expected value $\frac{i}{\eta-\lambda}$ and the total holding cost incurred during this time has expected value $\frac{hi}{\eta-\lambda}\big[\frac{1}{2}(i+1) + \frac{\lambda}{\eta-\lambda}\big]$, see e.g. Sect. 2.6 in Tijms [19].

Lemma 7. For $0 \leq i < i_1$,

$$\tau_{(i,1)}^E(R) = \frac{1}{(\lambda-\sigma_1\mu)^2}\left[(\lambda-\sigma_1\mu)(i_1-i) + \sigma_1\mu\big((\sigma_1\mu/\lambda)^{i_1} - (\sigma_1\mu/\lambda)^i\big)\right]$$

provided that $\lambda \neq \sigma_1\mu$; otherwise, $\tau_{(i,1)}^E(R) = \frac{1}{2\lambda}[i_1(i_1+1) - i(i+1)]$. Also, for $0 \leq i < i_1$,

$$\begin{aligned} c_{(i,1)}^E(R) = \frac{1}{(\lambda-\sigma_1\mu)^3}\Big[&\frac{1}{2}h(\lambda-\sigma_1\mu)^2(i_1^2-i^2) + (i_1-i)[r_1(\lambda-\sigma_1\mu)^2 \\ &-\frac{1}{2}h(\lambda+\sigma_1\mu)(\lambda-\sigma_1\mu)] + [-r_0(\lambda-\sigma_1\mu)^2 \\ &+r_1\lambda(\lambda-\sigma_1\mu) - h\lambda\sigma_1\mu]\left((\sigma_1\mu/\lambda)^{i_1} - (\sigma_1\mu/\lambda)^i\right)\Big] \end{aligned}$$

provided that $\lambda \neq \sigma_1\mu$; otherwise, $c_{(i,1)}^E(R) = \frac{1}{\lambda}\big[\frac{1}{6}h(i_1^3-i^3) + \frac{1}{2}r_1(i_1^2-i^2) + (r_0 - \frac{1}{2}r_1 - \frac{1}{6}h)(i_1-i)\big]$.

Proof. Put for abbreviation $\tau_i = \tau_{(i,1)}^E(R)$ and $c_i = c_{(i,1)}^E(R)$ for $0 \leq i < i_1$. By a conditioning argument,

$$\begin{aligned} \tau_i &= \frac{1}{\lambda+\sigma_1\mu} + \frac{\sigma_1\mu}{\lambda+\sigma_1\mu}\tau_{i-1} + \frac{\lambda}{\lambda+\sigma_1\mu}\tau_{i+1} \quad \text{for } 1 \leq i < i_1, \\ c_i &= \frac{hi+r_1}{\lambda+\sigma_1\mu} + \frac{\sigma_1\mu}{\lambda+\sigma_1\mu}c_{i-1} + \frac{\lambda}{\lambda+\sigma_1\mu}c_{i+1} \quad \text{for } 1 \leq i < i_1 \end{aligned}$$

with the boundary conditions $\tau_0 = 1/\lambda + \tau_1$, $c_0 = r_0/\lambda + c_1$ and $\tau_{i_1} = c_{i_1} = 0$. These equations are linear recurrence equations of the second order. Using a standard

method from the theory of linear difference equations, we get the desired expressions after tedious algebra.

We now specify how to compute the average cost $g(R)$ and the relative values $v_s(R)$ for the policy $R = (i_1, i_2)$. The set of embedded linear equations on E consists of two linear equations. Normalizing $v_{(i_2,2)}(R) = 0$ and using the results in Lemma 6, we obtain

$$\begin{aligned} g(R) &= \big[K + (\sigma_2\mu - \lambda)^{-1}(i_1 - i_2)\big(h(i_1 - i_2 + 1)/2 + h\lambda(\sigma_2\mu - \lambda)^{-1} \\ &\quad + r_2\big) + c^E_{(i_2,1)}(R)\big]\big[(\sigma_2\mu - \lambda)^{-1}(i_1 - i_2) + \tau^E_{(i_2,1)}(R)\big]^{-1}, \\ v_{(i_1,1)}(R) &= -c^E_{(i_2,1)}(R) + g(R)\tau^E_{(i_2,1)}(R), \end{aligned}$$

where $\tau^E_{(i_2,1)}(R)$ and $c^E_{(i_2,1)}(R)$ are given in Lemma 7. Next any other relative value can be obtained by a single-pass calculation. We have

$$\begin{aligned} &v_{(i,1)}(R) = c^E_{(i,1)}(R) - g(R)\tau^E_{(i,1)}(R) + v_{(i_1,1)}(R) \quad \text{for } 0 \le i < i_1, \\ &v_{(i,1)}(R) = K + v_{(i,2)}(R) \quad \text{for } i \ge i_1, \\ &v_{(i,2)}(R) = c^E_{(i,2)}(R) - g(R)\tau^E_{(i,2)}(R) + v_{(i_1,1)}(R) \quad \text{for } i > i_2, \\ &v_{(i,2)}(R) = v_{(i,1)}(R) \quad \text{for } 0 \le i \le i_2. \end{aligned}$$

Next we describe how to design the policy-improvement step such that the new policy has the same form as the current policy $R = (i_1, i_2)$. To do so, we must specify the policy-improvement quantity

$$T_R(s;a) = c_s(a) - g(R)\tau_s(a) + \sum_t p_{st}(a)v_t(R),$$

where the $p_{st}(a)$, $c_s(a)$ and $\tau_s(a)$ are the elements of the original semi-Markov decision model, and action $a = 1$ if transmission rate σ_1 is prescribed and $a = 2$ if transmission rate σ_2 is prescribed. Note that $T_R(s;a) = v_s(R)$ if $a = R_s$, since $g(R)$ and the $v_s(R)$ satisfy the original value-determination equations $v_s(R) = c_s(R_s) - g(R)\tau_s(R_s) + \sum_t p_{st}(R_s)v_t(R)$, as shown in Theorem 5. It is readily seen that

$$\begin{aligned} &T_R((i,1);1) = \tfrac{hi+r_1}{\lambda+\sigma_1\mu} - \tfrac{g(R)}{\lambda+\sigma_1\mu} + \tfrac{\sigma_1\mu}{\lambda+\sigma_1\mu}v_{(i-1,1)}(R) + \tfrac{\lambda}{\lambda+\sigma_1\mu}v_{(i+1,1)}(R),\ i \ge i_1, \\ &T_R((i,2);1) = T_R((i,1);1) \text{ for } i \ge i_1, \\ &T_R((i,1);2) = K + v_{(i,2)}(R) \text{ and } T_R((i,2);1) = v_{(i,1)}(R) \text{ for } i_2 < i < i_1, \\ &T_R((i,2);2) = \tfrac{hi+r_2}{\lambda+\sigma_2\mu} - \tfrac{g(R)}{\lambda+\sigma_2\mu} + \tfrac{\sigma_2\mu}{\lambda+\sigma_2\mu}v_{(i-1,2)}(R) + \tfrac{\lambda}{\lambda+\sigma_2\mu}v_{(i+1,2)}(R),\ i \le i_2. \end{aligned}$$

In the policy-improvement procedure we first adjust the switching level i_1 through a construction that guarantees that the policy-improvement quantity is strictly less than the relative value for any state $(i,1)$ with i between the current value of i_1 and the new value of i_1. Next we adjust the switching level i_2 in a similar way. This results in the following tailor-made algorithm that generates only policies of the (i_1, i_2)-type.

Algorithm

Step 0. Initialize with any (i_1, i_2) policy with $0 \leq i_2 < i_1$.
Step 1. For the current policy $R = (i_1, i_2)$, calculate the average cost $g(R)$ and the relative value $v_{(i_1,1)}(R)$ by solving the embedded system of linear equations with the normalization $v_{(i_2,2)}(R) = 0$. Any other relative value $v_s(R)$ needed in the following step is obtained by a single-pass calculation.
Step 2. (*a*) Search for the largest integer l_1 with $l_1 > i_1$ so that $T_R((i,1);1) < v_{(i,1)}(R)$ for $i_1 \leq i < l_1$. If such an integer l_1 exists, let $i_1^{new} = l_1$. Otherwise, search for the smallest integer m_1 with $i_2 < m_1 < i_1$ so that $T_R((i,1);1) < v_{(i,1)}(R)$ for $m_1 \leq i < i_1$. If such an integer exists, let $i_1^{new} = m_1$. Otherwise, $i_1^{new} = i_1$.
(*b*) Search for the smallest integer l_2 with $0 \leq l_2 < i_2$ so that $T_R((i,2);2) < v_{(i,2)}(R)$ for $l_2 < i \leq i_2$. If such an integer l_2 exists, let $i_2^{new} = l_2$. Otherwise, search for the largest integer m_2 with $i_2 < m_2 < i_1^{new}$ so that $T_R((i,2);1) < v_{(i,2)}(R)$ for $i_2 < i \leq m_2$. If such an integer exists, let $i_2^{new} = m_2$. Otherwise, $i_2^{new} = i_2$.
Step 3. If the new policy $R^{new} = (i_1^{new}, i_2^{new})$ is the same as the previous policy $R = (i_1, i_2)$, then the algorithm is stopped; otherwise, go to step 1 with R replaced by R^{new}.

This algorithm generates a sequence of (i_1, i_2) policies with non-increasing average costs. It is conjectured that the algorithm converges after finitely many steps to an (i_1, i_2) rule that is average cost optimal among the class of all conceivable control rules. In support of this conjecture are extensive numerical investigations. In all cases tested, we verified numerically that the average cost optimality equation was satisfied for the finally obtained (i_1, i_2) rule. Also, we found that the algorithm requires only a very few iterations. In general, policy iteration is empirically found to be a remarkably robust method that converges very fast in specific problems. The number of iterations is *practically independent* of the number of states and varies typically between 3 and 15 (say). Also, it can be roughly stated that the average costs of the policies generated by policy iteration converge at least exponentially fast to the minimum average cost, with the greatest improvements in the first few iterations. This empirical fact will be used in the analysis of the Markov decision application in the next section.

1.4 One-Step Policy Improvement for Suboptimal Policies

Policy iteration has the remarkable feature that it achieves the largest improvements in cost in the first few iterations. This finding underlies a heuristic approach for Markov decision problems with a multi-dimensional state space. In such decision problems it is usually not feasible to solve the value-determination equations. However, a policy-improvement step offers in general no computational difficulties. This suggests a heuristic approach that determines first a good estimate for the relative values and next applies only *one* policy-improvement step. By the na-

ture of the policy-iteration algorithm one might expect to obtain a good decision rule by the heuristic approach. How to compute the relative values to be used in the policy-improvement step typically depends on the specific application. The heuristic approach of attacking a multi-dimensional Markov decision problem through decomposition and a single-improvement step goes back to Norman [12] and has been successfully applied in Bhulai and Koole [7], Haijema and Van der Wal [6], Krishnan and Ott [10, 11], and Wijngaard [23], see also Powell [14]. This section deals with the dynamic routing model that was discussed in Krishnan and Ott [11]. In Sassen et al. [17] the heuristic approach has been applied to an extension of this dynamic routing model.

1.4.1 Dynamic Routing of Customers to Parallel Queues

An important queueing model arising in various practical situations is one in which arriving customers (messages or jobs) have to be assigned to one of several different groups of servers. Problems of this type occur in telecommunication networks and flexible manufacturing. The queueing system consists of n multi-server groups working in parallel, where each group has its own queue. There are s_k servers in group k, $(k = 1, \ldots, n)$. Customers arrive according to a Poisson process with rate λ. Upon arrival each customer has to be assigned to one of the n server groups. The assignment is irrevocable. The customer waits in the assigned queue until a server becomes available. Each server can handle only one customer at a time.

The problem is to find an assignment rule that (nearly) minimizes the average sojourn time per customer. This problem will be analyzed under the assumption that the service times of the customers are independent and exponentially distributed. The mean service time of a customer assigned to queue k is $1/\mu_k$ $(k = 1, \ldots, n)$. It is assumed that $\lambda < \sum_{k=1}^{n} s_k \mu_k$. In what follows we consider the minimization of the overall average number of customers in the system. In view of Little's formula, the minimization of the average sojourn time per customer is equivalent to the minimization of the average number of customers in the system.

The problem of assigning the arrivals to one of the server groups is a semi-Markov decision problem with a multi-dimensional state space. The decision epochs are the arrival epochs of new customers. The state of the system at a decision epoch is an n-dimensional vector $x = (i_1, \ldots, i_n)$, where i_j denotes the number of customers present in queue j. This description uses the memoryless property of the exponential service times. The action $a = k$ in state x means that the new arrival is assigned to queue k. To deal with the optimality criterion of the long-run average number of customers in the system, we impose the following cost structure on the system. A cost at rate j is incurred whenever there are j customers in the system. Then the long-run average cost per time unit gives the long-run overall average number of customers in the system.

An intuitively appealing control rule is the shortest-queue rule. Under this rule each arriving customer is assigned to the shortest queue. Except for the special case

of $s_1 = \ldots = s_n$ and $\mu_1 = \ldots = \mu_n$, this rule is in general not optimal. In particular, the shortest-queue rule may perform quite unsatisfactorily in the situation of a few fast servers and many slow servers. It is difficult to estimate the average cost and the relative values for the shortest-queue rule. Therefore this rule is not suited to base on the one-step policy improvement step. Another simple rule is the *Bernoulli-splitting rule*. Under this rule each arrival is assigned with a given probability p_k to queue k for $k = 1,\ldots,n$, irrespective of the queue lengths. This assignment rule produces independent Poisson streams at the various queues, where queue k receives a Poisson stream at rate λp_k. The probabilities p_k must satisfy $\sum_k p_k = 1$ and $\lambda p_k < s_k \mu_k$ for $k = 1,\ldots,n$. This condition guarantees that no infinitely long queues can build up. For the Bernoulli-splitting rule it is easy to find the average cost and the relative values. The reason is that under the Bernoulli rule the separate queues act as independent queues of the $M/M/s$ type and thus a natural decomposition of the problem is naturally obtained. In the $M/M/s$ queue with arrival rate α and s exponential servers each with service rate μ, the long-run average number of customers in the system equals

$$L(s,\alpha,\mu) = \frac{\rho(s\rho)^s}{s!(1-\rho)^2}\left\{\sum_{k=0}^{s-1}\frac{(s\rho)^k}{k!} + \frac{(s\rho)^s}{s!(1-\rho)}\right\}^{-1} + s\rho,$$

where $\rho = \alpha/(s\mu)$. Under the Bernoulli-splitting rule the overall average number of customers in the system equals

$$\sum_{k=1}^{n} L(s_k, \lambda p_k, \mu_k).$$

The best Bernoulli-splitting rule is found by minimizing this expression with respect to $p_1,\ldots,p_n$ subject to the condition $\sum_k p_k = 1$ and $0 \le \lambda p_k < s_k\mu_k$ for $k = 1,\ldots,n$. This minimization problem must be numerically solved by some search procedure (for the case of $n = 2$, bisection can be used to find the minimum of a unimodal function in a single variable).

One-Step Policy Improvement

Denote by policy $R^{(0)}$ the best Bernoulli-splitting rule and let $p_k^{(0)}$, $k = 1,\ldots,n$ be the splitting probabilities associated with policy $R^{(0)}$. We already pointed out that the average cost for rule $R^{(0)}$ is easy to compute. Below it will be shown that the relative values are also easy to obtain for rule $R^{(0)}$. Let us first explain how to derive an improved policy from the Bernoulli-splitting rule $R^{(0)}$. This derivation is based on first principles discussed in Sect. 6.2. The basic idea of the policy-improvement step is to minimize for each state x the difference $\Delta(x,a,R^{(0)})$ defined by

$$\Delta(x,a,R^{(0)}) = \text{the difference in total expected costs over an infinitely long period of time by taking first action } a \text{ and next using policy } R^{(0)} \text{ rather than using policy } R^{(0)} \text{ from scratch when the initial state is } x.$$

The difference is well-defined since the Markov chain associated with policy $R^{(0)}$ is aperiodic. Under the Bernoulli-splitting rule the n queues act as independent $M/M/s$ queues. Define for each separate queue k

$$D_k(i) = \text{the difference in total expected costs in queue } k \text{ over an infinitely long period of time by starting with } i+1 \text{ customers in queue } k \text{ rather than with } i \text{ customers.}$$

Then, for each state $x = (i_1, \ldots, i_n)$ and action $a = j$

$$\begin{aligned}\Delta(x,a,R^{(0)}) &= \sum_{\substack{k=1\\k\neq j}}^{n} p_k^{(0)}[-D_k(i_k) + D_j(i_j)] + p_j^{(0)} \times 0 \\ &= -\sum_{k=1}^{n} p_k^{(0)} D_k(i_k) + D_j(i_j).\end{aligned}$$

Since the term $\sum_k p_k^{(0)} D_k(i_k)$ does not depend on the action $a = j$, the step of minimizing $\Delta(x,j,R^{(0)})$ over j reduces to finding the minimizing index in

$$\min_{1\le j\le n} D_j(i_j).$$

Hence the expression to be evaluated in the policy-improvement step applied to the Bernoulli-splitting rule is remarkably simple. The suboptimal rule resulting from the single application of the policy-improvement step is called the *separable rule*. It remains to specify the function $D_k(i)$ for each queue k. To do so, consider an $M/M/s$ queue in isolation, where customers arrive according to a Poisson process with rate α and there are s exponential servers each with service rate μ. Each arrival is admitted to the queue. The state of the system describes the number of customers present. A cost at rate j is incurred when there are j customers present. The long-run average cost per time unit is given by

$$g = L(s,\alpha,\mu).$$

The $M/M/s$ queueing process can be seen as a semi-Markov decision process with a single decision in each state. The decision is to leave the system alone. In this Markov decision formulation it is convenient to consider the state of the system both at the arrival epochs and the service completion epochs. In the $M/M/s$ queue the situation of i customers present just after a service completion is probabilistically the same as the situation of i customers present just after an arrival. We define the relative cost function w_i

$$w(0) = 0 \text{ and } w(i) = K_i - gT_i \text{ for } i = 1, 2, \ldots,$$

where T_i is the expected time until the first return to an empty system starting with i customers present and K_i is the total expected cost incurred until the first return to an empty system starting with i customers present for $i \geq 1$. It suffices to have $w(0) = 0$ and there is no need to define T_0 and K_0 as the expected time and the expected total cost between two successive visits to state 0. By the interpretation of the relative values given in Sect. 1.1 (note that the underlying Markov chain is aperiodic), we have for any $i = 0, 1, \ldots$ that

$$\begin{aligned} w(i+1) - w(i) = {} & \text{the difference in total expected costs over an infinitely} \\ & \text{long period of time by starting in state } i+1 \text{ rather} \\ & \text{than in state } i. \end{aligned}$$

Denote by $w_k(i)$ the function $w(i)$ for the $M/M/s$ queue with $\alpha = \lambda p_k$, $s = s_k$, and $\mu = \mu_k$. The desired function $D_k(i)$ for queue k is given by

$$D_k(i) = w_k(i+1) - w_k(i).$$

The basic functions K_i and T_i are easy to compute. By conditioning,

$$\begin{aligned} T_i &= \frac{1}{\alpha + i\mu} + \frac{i\mu}{\alpha + i\mu} T_{i-1} + \frac{\alpha}{\alpha + i\mu} T_{i+1} \quad \text{for } 1 \leq i \leq s, \\ K_i &= \frac{i}{\alpha + i\mu} + \frac{i\mu}{\alpha + i\mu} K_{i-1} + \frac{\alpha}{\alpha + i\mu} K_{i+1} \quad \text{for } 1 \leq i \leq s, \end{aligned}$$

provided that we put $T_0 = K_0 = 0$. For $i > s$, we have

$$\begin{aligned} T_i &= \frac{i-s}{s\mu - \alpha} + T_s, \\ K_i &= \frac{i-s}{s\mu - \alpha} \left[\frac{1}{2}(i-s)(i-s+1) + \frac{\alpha}{s\mu - \alpha} \right] + \frac{s(i-s)}{s\mu - \alpha} + K_s. \end{aligned}$$

To see the latter relations, note that the time to reach an empty system from state $i > s$ is the sum of the time to reach state s and the time to reach an empty system from state s. By the memoryless property of the exponential distribution, the multi-server $M/M/s$ queue operates as a single-server $M/M/1$ queue with service rate $s\mu$ when s or more customers are present. Using the basic result for the $M/M/1$ queue stated in the proof of Lemma 6, we find the formulas for T_i and K_i when $i > s$. Substituting the expressions for T_{s+1} and K_{s+1} in the equations for T_i and K_i with $i = s$, we get two systems of linear equations for T_i, $1 \leq i \leq s$ and K_i, $1 \leq i \leq s$. Once these systems of linear equations have been solved, we can next compute T_i and K_i for any desired $i > s$.

Summarizing, the heuristic algorithm proceeds as follows.

Heuristic Algorithm

Step 1. Compute the best values $p_k^{(0)}$, $k = 1, \ldots, n$, of the Bernoulli-splitting probabilities by minimizing $\sum_{k=1}^{n} L(s_k, \lambda p_k, \mu_k)$ subject to $\sum_{k=1}^{n} p_k = 1$ and $0 \leq \lambda p_k < s_k \mu_k$ for $k = 1, \ldots, n$.
Step 2. For each queue $k = 1, \ldots, n$, solve the two systems of linear equations for the $T_i (= T_i(k))$ and the $K_i (= K_i(k))$ with $\alpha = \lambda p_k^{(0)}$, $s = s_k$ and $\mu = \mu_k$. Next compute for each queue k the average cost $g_k = L(s_k, \lambda p_k^{(0)}, \mu_k)$ and the function $w_k(i) = K_i(k) - g_k T_i(k)$.
Step 3. For each state $x = (i_1, \ldots, i_n)$, determine an index k_0 achieving the minimum in

$$\min_{1 \leq k \leq n} \{w_k(i_k + 1) - w_k(i_k)\}.$$

The separable rule assigns a new arrival in state $x = (i_1, \ldots, i_n)$ to queue k_0.

Let us consider the numerical data

$$s_1 = 10, \quad s_2 = 1, \quad \mu_1 = 1 \quad \text{and} \quad \mu_2 = 9.$$

The traffic load ρ, which is defined by

$$\rho = \lambda / (s_1 \mu_1 + s_2 \mu_2),$$

is varied as $\rho = 0.2, 0.5, 0.7, 0.8$ and 0.9. In addition to the theoretically minimum average sojourn time, Table 1.1 gives the average sojourn time per customer for the Bernoulli-splitting rule (B-split) and for the heuristic separable rule. The table also gives the average sojourn time per customer under the shortest-expected-delay (SED) rule. Under this rule an arriving customer is assigned to the queue in which its expected individual delay is smallest (in case of a tie, the customer is sent to queue 1). The results in the table show that this intuitively appealing control policy performs unsatisfactorily for the case of heterogenous services. However, the heuristic separable rule shows an excellent performance for all values of ρ.

Table 1.1: Numerical results for one-step policy improvement

ρ	SED	B-split	Separable	Optimal
0.2	0.192	0.192	0.191	0.191
0.5	0.647	0.579	0.453	0.436
0.7	0.883	0.737	0.578	0.575
0.8	0.982	0.897	0.674	0.671
0.9	1.235	1.404	0.941	0.931

1.5 One-Stage-Look-Ahead Rule in Optimal Stopping

Consider a process that is observed at time points $t = 0, 1, 2, \ldots$ to be in one of the states of a finite or countably infinite set I. In each state there are no more than two possible actions: $a = 0$ (stop) and $a = 1$ (continue). When in state i and choosing the stopping action $a = 0$, a terminal reward $R(i)$ is received. Suppose that there is a termination state that is entered upon choosing the stopping action. Once this state is entered the system stays in that state and no further costs or rewards are made thereafter. When in state i and choosing action $a = 1$, a continuation cost $c(i)$ is incurred and the process moves on to the next state according to the transition probabilities p_{ij} for $i, j \in I$. The following assumption is made.

Assumption. *Either the condition*

(i) $\sup_{i \in I} R(i) < \infty$ *and* $\inf_{i \in I} c(i) > 0$ *holds or the condition*

(ii) $\sup_{i \in I} R(i) < \infty$, $c(i) = 0$ *for* $i \in I$, *and there is a nonempty set* S_0 *consisting of the states in which stopping is mandatory and having the property that the process will reach the set* S_0 *within a finite expected time when always action* $a = 1$ *is chosen in the states* $i \notin S_0$, *whatever the starting state is.*

The goal is to find a stopping rule that minimizes the expected total costs over an unbounded planning horizon, interpreting the terminal reward $R(i)$ as a negative cost. This stopping model can be transformed into an equivalent model with nonnegative costs, see Ross [16]. To do so, let $R = \sup_{i \in I} R(i)$ and consider the equivalent process in which we pay R at the start, incur a nonnegative cost $c(i)$ when choosing action $a = 0$ in state i and incur the nonnegative terminal cost of $R - R(i)$ when stopping in state i, where the process moves on to state j with probability p_{ij} when action $a = 1$ is chosen in state i. In the equivalent model all costs are nonnegative so that results from the theory of negative dynamic programming apply. For the transformed process, let $V'(i)$ be the minimum expected total cost over an unbounded planning horizon when the process starts in state i and all conceivable policies are considered. The function $V'(i)$ satisfies the optimality equation $V'(i) = \min\{R - R(i), c(i) + \sum_{j \in I} p_{ij} V'(j)\}$ for $i \in I$ and the policy which chooses the minimizing actions is optimal. The cost function $V(i) = V'(i) - R$ is the minimum expected total cost in the original process and satisfies the optimality equation

$$V(i) = \min\Big\{ -R(i),\, c(i) + \sum_{j \in I} p_{ij} V(j) \Big\} \quad \text{for } i \in I.$$

In the case of Assumption(ii), $V(i) = -R(i)$ for $i \in S_0$ and the equation is only relevant for $i \notin S_0$. The stationary policy that chooses in each state i the action minimizing the right side of the optimality equation is optimal over the class of all conceivable policies. Moreover, the value function $V(i)$ is given by

$$V(i) = \lim_{n \to \infty} V_n(i) \quad \text{for } i \in I,$$

where $V_n(i)$ is obtained from $V_n(i) = \min\{-R(i), c(i) + \sum_{j\in I} p_{ij}V_{n-1}(j)\}$ starting with $V_0(i) = -R(i)$, see Ross [16]. It is also shown in Ross [16] that under a certain condition the so-called one-stage-look-ahead rule is optimal, see also the lucid class notes of Weber [22]. This rule is an intuitively appealing policy that looks only one step ahead, as the name says. Consider the set of states in which it is at least as good as to stop now as to continue one more step and then stop:

$$B = \{i \in I : R(i) \geq -c(i) + \sum_{j\in I} p_{ij}R(j)\}.$$

Clearly, it cannot be optimal to stop in a state $i \notin B$, since in that case it would be strictly better to continue one more step and then stop. The *one-stage-look-ahead rule* is now defined as the rule that stops in state i only if $i \in B$. The following theorem holds.

Theorem 8. *Suppose that the set B is closed so that once the process enters the set B it remains in B. That is, $p_{ij} = 0$ for $i \in B$ and $j \notin B$. Then the one-stage-look-ahead rule is optimal.*

A proof of this theorem was given in Ross [16] under Assumption (i), but an examination of the proof reveals that the theorem is also valid under Assumption (ii).

We give three interesting examples of optimal stopping problems. In the first example the closedness condition of Theorem 8 is satisfied so that the one-stage-look-ahead rule is optimal. In the other examples the closedness condition is not satisfied, but nevertheless the one-stage-look-ahead rule is very close in cost to the optimal policy. More examples of optimal stopping problems can be found in Boyce [2], Hill [8] and Tijms [20].

1.5.1 Devil's Penny Problem

Someone puts $n+1$ closed boxes in random order in front of you. One of these boxes contains a devil's penny and the other n boxes contain given dollar amounts $a_1, \ldots, a_n$. You may open as many boxes as you wish, but they must be opened one by one. You can keep the money from the boxes you have opened as long as you have not opened the box with the devil's penny. Once you open this box, the game is over and you lose all the money gathered so far. What is an optimal stopping rule when you want to maximize the expected value of your gain?

This problem can be put in the framework of the optimal stopping model. The state space is finite and given by

$$I = \{0\} \cup \{(k, b_1, \ldots, b_k) : 1 \leq k \leq n-1\} \cup \{(1,0)\}.$$

State 0 means that you have opened the box with the devil's penny. The state $(k, b_1, \ldots, b_k)$ means that there are still $k+1$ unopened boxes including the box

with the devil's penny, where dollar amounts $b_1,\ldots,b_k$ are contained in the k unopened boxes not having the devil's penny (each b_i is one of the $a_1,\ldots,a_n$). State $(1,0)$ means that the only unopened box is the box with the devil's penny. In state $(k,b_1,\ldots,b_k)$ you have gathered so far $A-\sum_{i=1}^{k} b_i$ dollars, where

$$A=\sum_{i=1}^{n} a_i$$

is the original total dollar amount in the n boxes. In state 0 the process stops and a terminal reward $R(0)=0$ is received. In the other states there are two possible actions $a=0$ and $a=1$. The action $a=1$ means that you continue and open one other box. There is no cost associated with this action. If you take the stopping action $a=0$ in state $s=(k,b_1,\ldots,b_k)$, you receive a terminal reward of $R(s)=A-\sum_{i=1}^{k} b_i$ (and $R(s)=A$ for $s=(1,0)$). Then the process moves on to state 0 with probability $\frac{1}{k+1}$ and to state $(k-1,b_1,\ldots,b_{l-1},b_{l+1},\ldots,b_k)$ with the same probability $\frac{1}{k+1}$ for $1\le l\le k$. To analyze the one-stage-look-ahead rule, let for any state $s=(k,b_1,\ldots,b_k)$ the random variable $D(s)$ denote the amount by which your current gain $G=A-\sum_{j=1}^{k} b_j$ would change when you would decide to open one other box when you are in state s. Then,

$$E\big(D(s)\big)=\frac{1}{k+1}\sum_{j=1}^{k} b_j-\frac{1}{k+1}G=\frac{1}{k+1}\Big(A-G\Big)-\frac{1}{k+1}G.$$

Obviously,

$$E\big(D(s)\big)\le 0 \quad \text{if and only } G\ge\frac{1}{2}A.$$

Let

$$B=\{0\}\cup\{s: E\big(D(s)\big)\le 0\}.$$

It is immediate that action $a=1$ in state $s\in B$ can only lead to states $t\in B$. Thus, we can conclude from Theorem 8 that the one-stage-look-ahead rule is optimal among the class of all conceivable control rules. The one-stage-look-ahead rule prescribes to stop in state $s=(k,b_1,\ldots,b_k)$ only if the dollar amount $G=A-\sum_{i=1}^{k} b_i$ gathered so far is at least half of the original total dollar amount A. It is quite remarkable that the optimal stopping rule depends only on the total dollar amount in the boxes and not on the distribution of the total dollar amount over the boxes. In the case that there are two boxes with a devil's penny, an examination of the analysis shows that it is optimal to stop as soon as the dollar amount you have gathered is at least one third of the original total dollar amount in the boxes.

1.5.2 A Game of Dropping Balls into Bins

A game machine can be used to drop balls into bins. The balls are dropped one at a time and any ball will land at random into one of b bins. You can stop dropping balls whenever you wish. At the end of the game you win \$1 for every bin with exactly one ball and you lose half of a dollar for every bin containing $k \geq 2$ balls. Empty bins do not count. What should you do when you want to maximize the expected net gain? How does change the solution when you lose $\frac{1}{2}k$ dollars rather than half of a dollar for every bin containing $k \geq 2$ balls?

These problems are optimal stopping problems. The first problem has a finite state space $I = \{(i_0, i_1) : i_0, i_1 \geq 0, i_0 + i_1 \leq b\}$, where state (i_0, i_1) means that there are i_0 empty bins and i_1 bins with exactly one ball (and $b - i_0 - i_1$ bins with two or more balls). For each state $s = (i_0, i_1)$, the continuation cost $c(s) = 0$ and the terminal reward $R(s) = i_1 - 0.5(b - i_0 - i_1)$. It is no restriction to impose the condition that stopping is mandatory in the states $(0, i_1)$. If you decide in state $s = (i_0, i_1)$ to drop one more ball and then stop rather than to stop now, then the expected change in your terminal reward is

$$\frac{i_0}{b} \times 1 - \frac{i_1}{b} \times (0.5 + 1) + \frac{b - i_0 - i_1}{b} \times 0.$$

Hence the set of states in which it is as least as good to stop now in state s as to continue for one more step and then stop is given by

$$B = \{(i_0, i_1) : i_0 - 1.5 i_1 \leq 0\}.$$

The one-stage-look-ahead rule prescribes to stop in the states of B and to continue otherwise. However, the one-stage-look-ahead rule is not the overall best control rule. For example, take $b = 3$. Then, calculations show that the overall best stopping rule differs only from the one-stage-look-ahead rule by the decision taken in state $(1, 1)$. The one-stage-look-ahead rule prescribes to stop in state $(1, 1)$. However, by taking the decision "not to stop" in state $(1, 1)$, there is a positive probability of going to state $(1, 0)$ outside the set B. In other words, the closedness condition in Theorem 8 is not satisfied. Nevertheless the performance of the one-stage-look-ahead rule is very good, as numerical investigations reveal. The expected gain under the one-stage-look-ahead rule can be calculated by a simple recursion. Let $u(i_0, i_1)$ be defined as the expected net gain you can achieve under the one-stage-look-ahead rule when starting in state (i_0, i_1). Then $u(i_0, i_1) = i_1 - 0.5(b - i_0 - i_1)$ for $i_0 \leq 1.5 i_1$. The desired $u(b, 0)$ can be obtained by applying the recursion

$$u(i_0, i_1) = \frac{i_0}{b} u(i_0 - 1, i_1 + 1) + \frac{i_1}{b} u(i_0, i_1 - 1) + \frac{b - i_0 - i_1}{b} u(i_0, i_1),$$

or, equivalently, $u(i_0, i_1) = \frac{i_0}{i_0 + i_1} u(i_0 - 1, i_1 + 1) + \frac{i_1}{i_0 + i_1} u(i_0, i_1 - 1)$. The optimal value function $v(i_0, i_1)$ being the maximum expected net gain you can achieve starting from state (i_0, i_1) can be obtained from the optimality equation

$$v(i_0,i_1) = \max\Big\{i_1 - 0.5(b-i_0-i_1),\ \frac{i_0}{b}\,v(i_0-1,i_1+1) + \frac{i_1}{b}\,v(i_0,i_1-1) + \frac{b-i_0-i_1}{b}\,v(i_0,i_1)\Big\},$$

or, equivalently, $v(i_0,i_1) = \max\{i_1 - 0.5(b-i_0-i_1), \overline{v}(i_0,i_1)\}$, where $\overline{v}(i_0,i_1)$ is given by

$$\overline{v}(i_0,i_1) = \frac{i_0}{i_0+i_1}v(i_0-1,i_1+1) + \frac{i_1}{i_0+i_1}v(i_0,i_1-1).$$

Since you always stop in state $(0,i_1)$, the boundary condition $v(0,i_1) = i_1 - 0.5(b-i_1)$ applies. The optimality equation can be solved by backwards calculations. First calculate $v(1,i_1)$ for $i_1 = 0,\ldots,b-1$. Next calculate $v(2,i_1)$ for $i_1 = 0,\ldots,b-2$. Continuing in this way, the desired value $v(b,0)$ is obtained. Numerical investigations lead to the conjecture that the optimal stopping rule has the following simple form:

> you stop only in the states (i_0,i_1) with $i_1 \le r$, where r is the smallest integer larger than or equal to $2i_0/3$.

In Table 1.2 we give for several values of b the expected net gain under the one-stage-look-ahead rule (OSLA) and the maximum expected net gain under the optimal rule. Also, we give the expected gain under the one-stage-look-ahead rule that prescribes stopping in the states (i_0,i_1) with $i_0 < 1.5i_1$ rather than $i_0 \le 1.5i_1$. It is remarkable how good the one-stage-look-ahead-rules perform, where OSLA$<$ is slightly better than OSLA.

Table 1.2: Expected net gain in the first balls-and-bins problem

b	OSLA	OSLA$<$	Optimal
3	1.5000	1.5000	1.5833
5	1.8500	2.0900	2.1171
10	3.3494	3.4557	3.4938
15	4.7605	4.8188	4.8550
25	7.5091	7.5377	7.5661
40	11.5860	11.6050	11.6246
70	19.7079	19.7220	19.7339
100	27.8226	27.8314	27.8400

The second problem has a countably infinite state space and this state space can be taken as $I = \{(i_0,i_1,k) : i_0,i_1,k \ge 0,\ i_0+i_1 \le b\}$, where state (i_0,i_1,k) means that there are i_0 empty bins and i_1 bins with exactly one ball and a total of k balls in the $b-i_0-i_1$ bins each containing two or more balls. Note that $k=0$ if $i_0+i_1=b$. For each state $s=(i_0,i_1,k)$, the continuation cost $c(s)=0$ and the terminal reward $R(s) = i_1 - 0.5k$. It is no restriction to impose the condition that stopping is mandatory in the states $(0,i_1,k)$. If you decide in state $s=(i_0,i_1,k)$ to drop one

more ball and then stop rather than to stop now, then the expected change in your terminal reward is

$$\frac{i_0}{b} \times 1 - \frac{i_1}{b} \times 2 - \frac{b - i_0 - i_1}{b} \times 0.5.$$

Hence the set of states in which it is as least as good to stop now as to continue for one more step and then stop is given by

$$B = \{(i_0, i_1, k) : 3i_0 - 3i_1 - b \leq 0\}.$$

The one-stage-look-ahead rule prescribes to stop in the states of B and to continue otherwise. The recursion scheme for the calculation of the expected net gain under the one-stage-look-ahead rule and the recursion scheme for the calculation of the optimal value function together with the optimal stopping rule are very similar to the recursion schemes above. However, the difficulty arises that the set of states is countably infinite because of the component k of the state (i_0, i_1, k). In numerical calculations this difficulty can be overcome by truncating the state space. A much better method to circumvent this difficulty was proposed by Jules Coret, a student at the University of Amsterdam. His observation was that the action prescribed in any state (i_0, i_1, k) with $k > 2(b - i_0 - i_1)$ is the same as the action prescribed in state (i_0, i_1, k) with $k = 2(b - i_0 - i_1)$, both for the one-stage-look-ahead rule and for the optimal rule. Letting $u(s)$ be the expected net gain under the one-stage-look-ahead rule and $v(s)$ be the maximum expected net gain under the optimal rule when starting from state s, it holds for $k > 2(b - i_0 - i_1)$ that

$$u(i_0, i_1, k) = u\big(i_0, i_1, 2(b - i_0 - i_1)\big) - \frac{1}{2}\big(k - 2(b - i_0 - i_1)\big),$$
$$v(i_0, i_1, k) = v\big(i_0, i_1, 2(b - i_0 - i_1)\big) - \frac{1}{2}\big(k - 2(b - i_0 - i_1)\big).$$

Thus we can formulate recursion schemes on a finite set of states in order to compute $u(b, 0, 0)$ and $v(b, 0, 0)$. For example, putting for abbreviation $u(i_0, i_1) = u(i_0, i_1, 2(b - i_0 - i_1))$, we have

$$u(i_0, i_1) = \frac{i_0}{b} u(i_0 - 1, i_1 + 1) + \frac{i_1}{b} u(i_0, i_1 - 1) + \frac{b - i_0 - i_1}{b} [u(i_0, i_1) - 0.5]$$

for the states with $3i_0 - 3i_1 - b > 0$. The boundary condition is $u(i_0, i_1) = i_1 - \frac{1}{2} \times 2(b - i_0 - i_1)$ for $3i_0 - 3i_1 - b \leq 0$.

Numerical results are given in Table 1.3. The one-stage-look-ahead rule OSLA$<$ prescribes stopping in the states (i_0, i_1, k) with $3i_0 - 3i_1 - b < 0$ rather than $3i_0 - 3i_1 - b \leq 0$ as in OSLA. We find again that the one-stage-look-ahead rules show an excellent performance, where OSLA$<$ is slightly better than OSLA.

Table 1.3: Expected net gain in the second balls-and-bins problem

b	OSLA	OSLA$<$	Optimal
3	1.0000	1.2500	1.2500
5	1.6333	1.6333	1.6333
10	2.6716	2.6716	2.6813
15	3.6596	3.7129	3.7129
25	5.7583	5.7583	5.7671
40	8.8467	8.8467	8.8532
70	15.0196	15.0196	15.0236
100	21.1902	21.1902	21.1931

1.5.3 The Chow-Robbins Game

An intriguing stopping problem is the Chow-Robbins game. You toss repeatedly a fair coin and you can stop whenever you want. Your payoff is the proportion of heads obtained at the time you stop. There is no limit on the number of tosses. What is an optimal rule for deciding when to stop and what is the expected value of your payoff under an optimal rule? This problem is very simple to state but very difficult to solve. Using the fact that the law of large numbers guarantees 50% heads in the long run, the lower bound $\frac{3}{4}$ for the maximal expected value of the payoff can be easily seen. This lower bound is achieved by the rule which prescribes to stop if the first toss results in heads, and otherwise to continue until the proportion of heads is $\frac{1}{2}$. In some states the decision whether to stop or go is easy to see but in general it is a challenging problem to find the best decisions. It has been proved that an optimal stopping rule exists and is characterized by integers $\beta_1, \beta_2, \ldots$ such that you stop after the nth toss when the number of heads minus the number of tails is larger than or equal to β_n. It is very difficult to compute the numerical values of the β_n and the precise value of the maximal expected payoff. The difficulty is that backward induction will not work for the optimality equation in the Chow-Robbins game. In Hägström and Wästlund [5], computer analysis was used to find the β_n for smaller values of n and it was shown that the maximum expected payoff is between the bounds $0.7929530\ldots$ and $0.7929556\ldots$. However, the simple idea of looking only one stage ahead leads to a remarkably simple heuristic whose expected payoff is very close to the maximum expected payoff. The analysis goes as follows. Suppose that the current proportion of heads after n tosses is f_n. If you decide to go for one more toss, then the expected value of the proportion of heads after $n+1$ tosses is

$$\frac{1}{2} \times \frac{nf_n + 1}{n+1} + \frac{1}{2} \times \frac{nf_n}{n+1}.$$

The inequality $\frac{nf_n+1}{2(n+1)} + \frac{nf_n}{2(n+1)} \geq f_n$ is equivalent to the inequality $f_n \leq \frac{1}{2}$. This leads to the simple stopping rule which prescribes to stop as soon the number of heads exceeds the number of tails. The expected payoff under this rule is $\frac{\pi}{4} \approx 0.785398\ldots$, which is very close the theoretically maximum expected payoff. A remarkable finding! The derivation of the value $\frac{\pi}{4}$ for the expected payoff of the heuristic rule is not difficult. We give an outline of the derivation. Define the random variable X as the number of tosses until the number of heads exceeds the number of tails for the first time, and let $p_n = P(X = n)$. Then the expected payoff under the simple stopping rule is

$$\sum_{k=1}^{\infty} \frac{k}{2k-1}\, p_{2k-1}.$$

The p_n can be obtained by the generating function approach. Using conditioning, we obtain that $P(z) = E(z^X)$ is given by $P(z) = \frac{1-\sqrt{1-z^2}}{z}$ for $|z| \leq 1$. Using the series expansion of $(1+x)^a$ with $a = 0.5$ and $x = -z^2$, we find that $P(z)$ can be evaluated as $P(z) = \sum_{k=1}^{\infty} \left(\frac{1}{2}\right)^{2k-1} \frac{(2k-2)!}{k!(k-1)!}\, z^{2k-1}$. This gives $p_{2k-1} = (\frac{1}{2})^{2k-1} \frac{(2k-2)!}{k!(k-1)!}$ for $k \geq 1$ and so the expected value of the payoff is

$$\sum_{k=1}^{\infty} \frac{k}{2k-1}\, p_{2k-1} = \frac{1}{2} \sum_{k=1}^{\infty} \frac{1}{2k-1} \left(\frac{1}{2}\right)^{2k-2} \binom{2k-2}{k-1}.$$

Using the Taylor series expansion of $\arcsin(x)$, we next find the desired result that the expected reward is equal to $\frac{1}{2}\arcsin(1) = \frac{\pi}{4}$.

References

1. S. Bhulai, G. Koole, On the structure of value functions for threshold policies in queueing models. J. Appl. Probab. **40**, 613–622 (2003)
2. W.M. Boyce, On a simple stopping problem. Discret. Math. **5**, 297–312 (1973)
3. E.V. Denardo, *Dynamic Programming* (Prentice-Hall, Englewood Cliffs, NJ, 1980)
4. C. Derman, *Finite State Markovian Decision Processes* (Academic, New York, 1970)
5. O. Hägström, J. Wästlund, Rigorous computer analysis of the Chow-Robbins game. Am. Math. Mon. **120**, 893–900 (2013)
6. R. Haijema, J. Van der Wal, An MDP decomposition approach for traffic control at isolated signalized intersections. Probab. Eng. Inf. Sci. **27**, 587–602 (2008)
7. N.A.J. Hastings, Bounds on the gain of a Markov decision process. Oper. Res. **19**, 240–244 (1971)
8. T.P. Hill, Knowing when to stop. Am. Sci. **97**, 126–133 (2007)
9. R.A. Howard, *Dynamic Programming and Markov Processes* (Wiley, New York, 1960)

10. K.R. Krishnan, T.J. Ott, State-dependent routing for telephone traffic: theory and results, in *Proceedings of 25th IEEE Conference on Decision and Control*, Athens (IEEE, New York, 1986), pp. 2124–2128
11. K.R. Krishnan, T.J. Ott, Joining the right queue: a Markov decision rule, in *Proceedings of 26th IEEE Conference on Decision and Control*, Los Angeles, CA (IEEE, New York, 1987), pp. 1863–1868
12. J.M. Norman, *Heuristic Procedures in Dynamic Programming* (Manchester University Press, Manchester, 1972)
13. A. Odoni, On finding the maximal gain for Markov decision processes. Operat. Res. **17**, 857–860 (1969)
14. W.B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality* (Wiley, New York, 2007)
15. M.L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (Wiley, New York, 1994)
16. S.M. Ross, *Introduction to Stochastic Dynamic Programming*, (Academic, New York, 1983)
17. S.A.E. Sassen, H.C. Tijms, R.D. Nobel, A heuristic rule for routing customers to parallel servers. Statistica Neerlandica **51**, 107–121 (1997)
18. P.J. Schweitzer, A. Federgruen, Geometric convergence of value iteration in multichain Markov decision problems. Adv. Appl. Probab. **11**, 188–217 (1979)
19. H.C. Tijms, *A First Course in Stochastic Models* (Wiley, New York, 2003)
20. H.C. Tijms, *Understanding Probability*, 3rd edn. (Cambridge University Press, New York, 2012)
21. J. Van der Wal, The method of value oriented successive approximations for the average reward Markov decision process. OR Spektrum **1**, 233–242 (1980)
22. R. Weber, *Optimization and Control*. Class Notes (University of Cambridge, Cambridge, 2014). http://www.statslab.cam.ac.uk/rrw1/oc/oc2014.pdf
23. J. Wijngaard, Decomposition for dynamic programming in production and inventory control. Eng. Process Econ. **4**, 385–388 (1979)

Chapter 2
Value Function Approximation in Complex Queueing Systems

Sandjai Bhulai

Abstract The application of Markov decision theory to the control of queueing systems often leads to models with enormous state spaces. Hence, direct computation of optimal policies with standard techniques and algorithms is almost impossible for most practical models. A convenient technique to overcome this issue is to use one-step policy improvement. For this technique to work, one needs to have a good understanding of the queueing system under study, and its (approximate) value function under policies that decompose the system into less complicated systems. This warrants the research on the relative value functions of simple queueing models, that can be used in the control of more complex queueing systems. In this chapter we provide a survey of value functions of basic queueing models and show how they can be applied to the control of more complex queueing systems.

Key words: One-step policy improvement, Relative value function, Complex queueing systems, Approximate dynamic programming

2.1 Introduction

In its simplest form, a queueing system can be described by customers arriving for service, waiting for service if it cannot be provided immediately, and leaving the system after being served. The term customer is used in a general sense here, and does not necessarily refer to human customers. The performance of the system is usually measured on the basis of throughput rates or the average time customers

S. Bhulai (✉)
Faculty of Sciences, Vrije Universiteit Amsterdam, De Boelelaan 1081a,
1081 HV Amsterdam, The Netherlands
e-mail: s.bhulai@vu.nl

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_2

remain in the system. Hence, the average cost criterion is usually the preferred criterion in queueing systems (see, e.g., [18, 23]).

It is hardly necessary to emphasize the applicability of theory on queueing systems in practice, since many actual queueing situations occur in daily life. During the design and operation of such systems, many decisions have to be made; think of the availability of resources at any moment in time, or routing decisions. This effect is amplified by the advances in the field of data and information processing, and the growth of communication networks (see, e.g., [1]). Hence, there is a need for optimal control of queueing systems.

In general, the control of queueing systems does not fit into the framework of discrete time Markov decision problems, in which the time between state transitions is fixed. When the time spent in a particular state follows an arbitrary probability distribution, it is natural to allow the decision maker to take actions at any point in time. This gives rise to decision models in which the system evolution is described continuously in time, and in which the cost accumulates continuously in time as well. If the cost function is independent of the time spent in a state, such that it only depends on the state and action chosen at the last decision epoch, then we can restrict our attention to models in which the decision epochs coincide with the transition times. Moreover, if the time spent between decision epochs follows an exponential distribution, then the queueing system under study can be reformulated as a discrete-time Markov decision problem. This discretization technique, known as uniformization, is due to [12], and was later formalized by Serfozo [21].

After uniformization, Markov decision theory can, in principle, be applied to the control of queueing systems, and usually gives rise to infinite state spaces with unbounded cost functions. In this setting, value iteration and policy iteration algorithms can be used to derive optimal policies. However, these algorithms require memory storage of a real-valued vector of at least the size of the state space. For denumerably infinite state spaces this is not feasible, and one has to rely on appropriate techniques to bound the state space (see, e.g., [8]). But even then, memory may be exhausted by the size of the resulting state space; this holds even more for multi-dimensional models. This phenomenon, known as the curse of dimensionality (see [4]), gives rise to high-dimensional systems and calls for approximation methods to the optimal policies.

A first approximation method can be distilled from the policy iteration algorithm. Suppose that one fixes a policy which, while perhaps not optimal, is not totally unreasonable either. This policy is evaluated through analytically solving the Poisson equations induced by the policy in order to obtain the long-run expected average cost and the average cost value function under this policy. Next, using these expressions the policy can be improved by doing one policy improvement step. We know that this policy is better, i.e., has a smaller or equal (if already optimal) average cost. It must be expected that the improved policy is sufficiently complicated to render another policy evaluation step impossible.

The idea of applying one-step policy improvement goes back to [15]. It has been successfully applied by Ott and Krishnan [17] for deriving state-dependent routing schemes for high-dimensional circuit-switched telephone networks. The initial

policy in their system was chosen such that the communication lines were independent. In that case, the value function of the system is the sum of the value functions for the independent lines, which are easier to derive. Since then, this idea has been used in many papers, see, e.g., [10, 19, 20, 25]. The initial policy in these papers was chosen such that the queues were independent, reducing the analysis to single queues. For this purpose, [5, 6] presented a unified approach to obtain the value function of the various queueing systems for various cost structures.

In this chapter, we review the theory and applications of the relative value functions of the most fundamental queueing systems: the M/Cox(r)/1 single-server queue, the M/M/s multi-server queue, the M/M/s/s blocking system, and a priority queueing system. Each of these systems has its own distinguishing characteristics. The M/Cox(r)/1 queue is a very general single-server queue that allows for approximations to single-server systems with non-exponential service distributions. Both the M/M/s and the M/M/s/s multi-server queues typically occur in many service systems such as telecommunication systems, call centers, cash registers, etc. The priority queueing system is an intriguing example in which the queues are not independent, and this is reflected as well in the relative value function.

The different characteristics of the queueing systems that we study will be illustrated by three examples of controlled queueing systems. In the first example, we consider the problem of routing customers to parallel single-server queues, where each server has its own general service time distribution. We demonstrate how the M/Cox(r)/1 queue can approximate the single-server queues after which one-step of policy improvement leads to nearly optimal policies. In the second example, we study a skill-based routing problem in a call center. This is a highly complex problem in which agent groups in the system are highly dependent on each others states. We illustrate how the value function of the multi-server queue can be adjusted to take these dependencies into account. In the last example, we focus on a controlled polling system. Here the value function of the priority queueing system can be used to take actions that directly consider the dependencies between the states of the different queues.

The remainder of the chapter is structured as follows. In Sect. 2.2 we develop difference calculus that can be used to obtain the relative value function of most one-dimensional queueing systems. We proceed with the survey of the relative value functions of the fundamental queueing systems in Sect. 2.3 and list all relevant special cases as well. We then illustrate the use of these value functions through three examples in Sect. 2.4 on routing to parallel queues, Sect. 2.5 on skill-based routing in call centers, and Sect. 2.6 on a controlled polling model, respectively.

2.2 Difference Calculus for Markovian Birth-Death Systems

In this section we will study the relative value function for a Markovian birth-death queueing system. We shall derive a closed-form expression for the long-run expected average costs and the relative value function by solving the Poisson equa-

tions. The Poisson equations give rise to linear difference equations. Due to the nature of the Poisson equations, the difference equations have a lot of structure when the state description is one-dimensional. Therefore, it is worthwhile to study difference equations prior to the analysis of the birth-death queueing system. We shall restrict attention to second-order difference equations, since this is general enough to model birth-death queueing processes.

Let $V(x)$ be an arbitrary function defined on $\mathbb{N}_0$. Define the backward difference operator Δ as

$$\Delta V(x) = V(x) - V(x-1),$$

for $x \in \mathbb{N}$. Note that the value of $V(x)$ can be expressed as

$$V(x) = V(k-1) + \sum_{i=k}^{x} \Delta V(i), \tag{2.1}$$

for every $k \in \mathbb{N}$ such that $k \leq x$. This observation is key to solving first-order difference equations, and second-order difference equations when one solution to the homogeneous equation is known. We first state the result for first-order difference equations. The result can be found in Chap. 2 of Mickens [14], but is stated less precise there.

Lemma 2.1. *Let $f(x)$ be a function defined on $\mathbb{N}$ satisfying the relation*

$$f(x+1) - \gamma(x) f(x) = r(x), \tag{2.2}$$

with γ and r arbitrary functions defined on $\mathbb{N}$ such that $\gamma(x) \neq 0$ for all $x \in \mathbb{N}$. With the convention that an empty product equals one, $f(x)$ is given by

$$f(x) = f(1)\,Q(x) + Q(x) \sum_{i=1}^{x-1} \frac{r(i)}{Q(i+1)}, \quad \textit{with} \quad Q(x) = \prod_{i=1}^{x-1} \gamma(i).$$

Proof. Let the function $Q(x)$ be as stated in the theorem. By assumption we have that $Q(x) \neq 0$ for all $x \in \mathbb{N}$. Dividing Eq. (2.2) by $Q(x+1)$ yields

$$\Delta \left[\frac{f(x+1)}{Q(x+1)} \right] = \frac{f(x+1)}{Q(x+1)} - \frac{f(x)}{Q(x)} = \frac{r(x)}{Q(x+1)}.$$

From Expression (2.1) with $k = 2$ it follows that

$$f(x) = Q(x) \frac{f(1)}{Q(1)} + Q(x) \sum_{i=2}^{x} \frac{r(i-1)}{Q(i)} = f(1)\,Q(x) + Q(x) \sum_{i=1}^{x-1} \frac{r(i)}{Q(i+1)}.$$

Note that the condition $\gamma(x) \neq 0$ for all $x \in \mathbb{N}$ is not very restrictive in practice. If there is a state $y \in \mathbb{N}$ for which $\gamma(y) = 0$, then the analysis can be reduced to two other first-order difference equations, namely the part for states $x < y$, and the part for states $x > y$ for which $\gamma(y) = 0$ is the boundary condition.

The solution to first-order difference equations plays an important part in solving second-order difference equations when one solution to the homogeneous equation is known. In that case, the second-order difference equation can be reduced to a first-order difference equation expressed in the homogeneous solution. Application of Lemma 2.1 then gives the solution to the second-order difference equation. The following theorem summarizes this result.

Theorem 2.1. *Let $V(x)$ be a function defined on $\mathbb{N}_0$ satisfying the relation*

$$V(x+1)+\alpha(x)V(x)+\beta(x)V(x-1)=q(x), \tag{2.3}$$

with α, β, and q arbitrary functions defined on $\mathbb{N}$ such that $\beta(x)\neq 0$ for all $x\in\mathbb{N}$. Suppose that one homogeneous solution is known, say $V_1^h(x)$, such that $V_1^h(x)\neq 0$ for all $x\in\mathbb{N}_0$. Then, with the convention that an empty product equals one, $V(x)$ is given by

$$\frac{V(x)}{V_1^h(x)}=\frac{V(0)}{V_1^h(0)}+\left[\Delta\left[\frac{V(1)}{V_1^h(1)}\right]\right]\sum_{i=1}^{x}Q(i)+\sum_{i=1}^{x}Q(i)\sum_{j=1}^{i-1}\frac{q(j)}{V_1^h(j+1)\,Q(j+1)},$$

where $Q(x)=\prod_{i=1}^{x-1}\beta(i)\,V_1^h(i-1)/V_1^h(i+1)$.

Proof. Note that V_1^h always exists, since a second-order difference equation has exactly two linearly independent homogeneous solutions (see Theorem 3.11 of Mickens [14]). The known homogeneous solution $V_1^h(x)$ satisfies

$$V_1^h(x+1)+\alpha(x)V_1^h(x)+\beta(x)V_1^h(x-1)=0. \tag{2.4}$$

Set $V(x)=V_1^h(x)\,u(x)$ for an arbitrary function u defined on $\mathbb{N}_0$. Substitution into Eq. (2.3) yields

$$V_1^h(x+1)\,u(x+1)+\alpha(x)V_1^h(x)\,u(x)+\beta(x)V_1^h(x-1)\,u(x-1)=q(x).$$

By subtracting Eq. (2.4) $u(x)$ times from this expression, and rearranging the terms, we derive

$$\Delta u(x+1)-\gamma(x)\Delta u(x)=r(x),$$

with

$$\gamma(x)=\frac{V_1^h(x-1)}{V_1^h(x+1)}\beta(x),\quad\text{and}\quad r(x)=\frac{q(x)}{V_1^h(x+1)}.$$

From Lemma 2.1 it follows that

$$\Delta u(x)=[\Delta u(1)]\,Q(x)+Q(x)\sum_{j=1}^{x-1}\frac{r(j)}{Q(j+1)},\quad\text{with}\quad Q(x)=\prod_{i=1}^{x-1}\gamma(i).$$

From Expression (2.1) it finally follows that

$$u(x) = u(0) + [\Delta u(1)] \sum_{i=1}^{x} Q(i) + \sum_{i=1}^{x} Q(i) \sum_{j=1}^{i-1} \frac{r(j)}{Q(j+1)}.$$

Since $V(x) = V_1^h(x)\, u(x)$ it follows that $u(x) = V(x)/V_1^h(x)$ for $x = 1, 2$.

From Theorem 3.11 of Mickens [14] it follows that a second-order difference equation has exactly two homogeneous solutions that are also linearly independent. In Theorem 2.1 it is assumed that one homogeneous solution V_1^h is known. From the same theorem it follows that the second homogeneous solution is determined by $V_2^h(x) = V_1^h(x) \sum_{i=1}^{x} Q(i)$. This can be easily checked as follows.

$$V_2^h(x+1) + \alpha(x) V_2^h(x) + \beta(x) V_2^h(x-1) =$$
$$V_1^h(x+1) \sum_{i=1}^{x+1} Q(i) + \alpha(x) V_1^h(x) \sum_{i=1}^{x} Q(i) + \beta(x) V_1^h(x-1) \sum_{i=1}^{x-1} Q(i) =$$
$$\sum_{i=1}^{x} Q(i) \Big[V_1^h(x+1) + \alpha(x) V_1^h(x) + \beta(x) V_1^h(x-1) \Big] +$$
$$V_1^h(x+1)\, Q(x+1) - \beta(x) V_1^h(x-1)\, Q(x) = 0.$$

The last equality follows from the fact that

$$Q(x+1) = \frac{V_1^h(x-1)}{V_1^h(x+1)} \beta(x)\, Q(x) = \frac{V_1^h(0)\, V_1^h(1)}{V_1^h(x)\, V_1^h(x+1)} \prod_{i=1}^{x} \beta(i), \quad x \in \mathbb{N}.$$

Note that the homogeneous solutions V_1^h and V_2^h are also linearly independent, since their Casorati determinant $C(x) = V_1^h(x)\, V_1^h(x+1)\, Q(x+1)$ is non-zero for all $x \in \mathbb{N}_0$ (see Sects. 3.2 and 3.3 of Mickens [14]).

With the use of Theorem 2.1 as a tool, we are ready to study birth-death queueing systems. The name birth-death stems from the fact that arrivals (birth) and departures (death) of customers occur only in sizes of one, i.e., batch arrivals and batch services are not allowed. Let state $x \in \mathcal{X} = \mathbb{N}_0$ denote the number of customers in the system. When the system is in state x, customers arrive according to a Poisson process with rate $\lambda(x)$. At the same time, customers receive service with an exponentially distributed duration at rate $\mu(x)$, such that $\mu(0) = 0$. Moreover, the system is subject to costs $c(x)$ in state x, where $c(x)$ is a polynomial function of x.

For stability of the system, we assume that

$$0 < \liminf_{x \to \infty} \frac{\lambda(x)}{\mu(x)} \leq \limsup_{x \to \infty} \frac{\lambda(x)}{\mu(x)} < 1.$$

Moreover, we assume that the transition rates satisfy

$$0 < \inf_{x \in \mathbb{N}_0} \big(\lambda(x) + \mu(x) \big) \leq \sup_{x \in \mathbb{N}_0} \big(\lambda(x) + \mu(x) \big) < \infty.$$

Without loss of generality we can assume that $\sup_{x\in\mathbb{N}_0}\big(\lambda(x)+\mu(x)\big)<1$. This can always be obtained after a suitable renormalization without changing the long-run system behavior. After uniformization, the resulting birth-death process has the following transition rate matrix.

$$\begin{array}{lll} & P_{0,0}=1-\lambda(0), & P_{0,1}=\lambda(0), \\ P_{x,x-1}=\mu(x), & P_{x,x}=1-\lambda(x)-\mu(x), & P_{x,x+1}=\lambda(x), \qquad x=1,2,\ldots. \end{array}$$

Inherent to the model is that the Markov chain has a unichain structure. Hence, we know that the long-run expected average cost g is constant. Furthermore, the value function V is unique up to a constant. As a result we can use this degree of freedom to set $V(0)=0$. Then, the Poisson equations for this system are given by

$$g+\big(\lambda(x)+\mu(x)\big)V(x)=\lambda(x)V(x+1)+\mu(x)V\big([x-1]^+\big)+c(x), \tag{2.5}$$

for $x\in\mathbb{N}_0$, where $[x]^+=\max\{0,x\}$. When the state space $\mathcal{X}$ is not finite, as is the case in our model, it is known that there are many pairs of the average cost g and the corresponding value function V that satisfy the Poisson equations (see, e.g., [7]). There is only one pair that is the correct solution, however, which we refer to as the unique solution. Finding this pair involves constructing a weighted norm such that the Markov chain is geometrically recurrent with respect to that norm. Next, this weighted norm poses extra conditions on the solution to the Poisson equations such that the unique solution to the Poisson equations can be obtained. Solving the Poisson equations therefore requires two steps; first one has to find an expression that satisfies the Poisson equations (existence), then one has to show that it is the unique solution (uniqueness).

In order to show the existence of solutions to the Poisson equations, we use the relation with the difference calculus by rewriting the Poisson equations in the form of Eq. (2.3), with

$$\alpha(x)=-\frac{\lambda(x)+\mu(x)}{\lambda(x)}, \quad \beta(x)=\frac{\mu(x)}{\lambda(x)}, \quad \text{and} \quad q(x)=\frac{g-c(x)}{\lambda(x)}.$$

Observe that $\beta(x)\neq 0$ for $x\in\mathbb{N}$ due to the stability assumption. Furthermore, note that for any set of Poisson equations, the constant function is always a solution to the homogeneous equations. This follows directly from the fact that P is a transition probability matrix. Hence, by applying Theorem 2.1 with $V_1^h(x)=1$ for $x\in\mathbb{N}_0$, it follows that the solution to Eq. (2.5) is given by

$$V(x)=\frac{g}{\lambda(0)}\sum_{i=1}^{x}Q(i)+\sum_{i=1}^{x}Q(i)\sum_{j=1}^{i-1}\frac{q(j)}{Q(j+1)}, \quad \text{with} \quad Q(x)=\prod_{i=1}^{x-1}\frac{\mu(i)}{\lambda(i)}.$$

We know that this expression is not the unique solution to the Poisson equations (see, e.g., [22]). Hence, we need to construct a weighted norm w such that the Markov chain is w-geometrically recurrent with respect to a finite set $M\subset\mathbb{N}_0$ to

show uniqueness (see Chap. 2 of Spieksma [22]), i.e., there should be an $\varepsilon > 0$ such that $\|{}_M P\|_w \leq 1 - \varepsilon$, where

$$ {}_M P_{ij} = \begin{cases} P_{ij}, & j \notin M, \\ 0, & j \in M, \end{cases} $$

and

$$ \|A\|_w = \sup_{i \in \mathcal{X}} \frac{1}{w(i)} \sum_{j \in \mathcal{X}} |A_{ij}| \, w(j). $$

Due to stability of the Markov chain there exists a constant $K \in \mathbb{N}_0$ such that $\lambda(x)/\mu(x) < 1$ for all $x > K$. Let $M = \{0, \ldots, K\}$, and assume that $w(x) = z^x$ for some $z > 1$. Now consider

$$ \sum_{y \notin M} \frac{P_{xy} w(y)}{w(x)} = \begin{cases} \lambda(K) z, & x = K, \\ \lambda(K+1) z + \big(1 - \lambda(K+1) - \mu(K+1)\big), & x = K+1, \\ \lambda(x) z + \big(1 - \lambda(x) - \mu(x)\big) + \frac{\mu(x)}{z}, & x > K+1. \end{cases} $$

We need to choose z such that all expressions are strictly less than 1. The first expression immediately gives $z < 1/\lambda(K)$. The second expression gives that $z < 1 + \mu(K+1)/\lambda(K+1)$. Solving the third expression shows that $1 < z < \mu(x)/\lambda(x)$ for $x > K+1$. Define $z^* = \min\{1/\lambda(K), \inf_{x \in \mathbb{N} \setminus M} \mu(x)/\lambda(x)\}$, and note that the choice of M ensures us that $z^* > 1$. Then, the three expressions are strictly less than 1 for all $z \in (1, z^*)$. Thus, we have shown that for $w(x) = z^x$ with $1 < z < z^*$, there exists an $\varepsilon > 0$ such that $\|{}_M P\|_w \leq 1 - \varepsilon$. Hence, the Markov chain is w-geometrically recurrent with respect to M.

Note that c is bounded with respect to the supremum norm weighted by w, since $c(x)$ is a polynomial in x by assumption. We know that the unique value function V is bounded with respect to the norm w (Chap. 2 of Spieksma [22]). Therefore, the value function cannot contain terms θ^x with $\theta > 1$, since the weight function can be chosen such that $z \in (1, \min\{\theta, z^*\})$. Consequently, $\|V\|_w = \infty$, hence the value function cannot grow exponentially with a growth factor greater than 1. Thus, we have the following corollary.

Corollary 2.1. *The value function of a stable birth-death queueing process cannot grow exponentially with a growth factor greater than 1.*

2.3 Value Functions for Queueing Systems

In this section, we provide the relative value functions of the most fundamental queueing systems: the M/Cox(r)/1 single-server queue with its special cases, the M/M/s multi-server queue, the M/M/s/s blocking system, and a priority queueing system. The value functions will be used in the examples later to illustrate their effectiveness in the control of complex queueing systems.

2.3.1 The M/Cox(r)/1 Queue

Consider a single server queueing system to which customers arrive according to a Poisson process with parameter λ. The service times are independent identically distributed and follow a Coxian distribution of order r. Thus, the service of a customer can last up to r exponential phases. The mean duration of phase i is μ_i for $i = 1,\dots,r$. The service starts at phase 1. After phase i the service ends with probability $1-p_i$, or it enters phase $i+1$ with probability p_i for $i = 1,\dots,r-1$. The service is completed with certainty after phase r, if not completed at an earlier phase. We assume that $p_i > 0$ for $i = 1,\dots,r-1$ to avoid trivial situations. Let $\mathcal{X} = \{(0,0)\} \cup \mathbb{N} \times \{0,\dots,r-1\}$ denote the state space, where for $(x,y) \in \mathcal{X}$ the component x represents the number of customers in the system, and y the number of completed phases of the service process.

Assume that the system is subject to costs for holding a customer in the system. Without loss of generality we assume that unit costs are incurred for holding a customer per unit of time in the system. Let $u_t(x)$ denote the total expected cost up to time t when the system starts in state x. Let $\gamma(i) = \prod_{k=1}^{i} p_k$ for $i = 0,\dots,r-1$ with the convention that $\gamma(0) = 1$. Note that the Markov chain satisfies the unichain condition. Assume that the stability condition

$$\sum_{k=1}^{r} \gamma(k-1)\frac{\lambda}{\mu_k} = \sum_{k=1}^{r}\prod_{l=1}^{k-1} p_l \frac{\lambda}{\mu_k} < 1,$$

holds, such that, consequently, the average cost $g = \lim_{t\to\infty} u_t(x)/t$ is independent of the initial state x due to Proposition 8.2.1 of Puterman [18]. Therefore, the dynamic programming optimality equations for the M/Cox(r)/1 queue are given by

$$\begin{aligned}
g + \lambda V(0,0) &= \lambda V(1,0), \\
g + (\lambda+\mu_i)V(x,i-1) &= \lambda V(x+1,i-1) + p_i\mu_i V(x,i) + \\
&\quad (1-p_i)\mu_i V(x-1,0) + x, \qquad i = 1,\dots,r-1, \\
g + (\lambda+\mu_r)V(x,r-1) &= \lambda V(x+1,r-1) + \mu_r V(x-1,0) + x.
\end{aligned}$$

The solution to this set of equations is given in the following theorem.

Theorem 2.2 (Theorem 1 in [5]). *Let $\gamma(i) = \prod_{k=1}^{i} p_k$ for $i = 0,\dots,r-1$ with the convention that $\gamma(0) = 1$. Define*

$$\alpha = \frac{\sum_{k=1}^{r} \frac{\gamma(k-1)}{\mu_k}}{1 - \sum_{k=1}^{r} \frac{\gamma(k-1)}{\mu_k}\lambda}, \text{ and } a_0 = \sum_{k=1}^{r} \frac{1-\gamma(k-1)}{\mu_k}\lambda\alpha - \sum_{k=1}^{r}\sum_{l=1}^{k-1} \frac{\gamma(l-1)}{\mu_k\mu_l}\lambda(1+\lambda\alpha).$$

The solution to the Poisson equations is then given by the average cost $g = \lambda(\alpha + a_0)$ and the corresponding value function

$$V(x,y) = \alpha \frac{x(x+1)}{2} + \left[a_0 + \left[\frac{1}{\gamma(y)} - 1 \right] \alpha - \sum_{k=1}^{y} \frac{\gamma(k-1)}{\gamma(y)} \frac{1+\lambda\alpha}{\mu_k} \right] x$$

$$- \left[a_0 + \sum_{k=y+1}^{r} \frac{\gamma(k-1)}{\gamma(y)} \frac{\lambda}{\mu_k} \left(\sum_{l=1}^{k-1} \frac{\gamma(l-1)}{\gamma(k-1)} \frac{1+\lambda\alpha}{\mu_l} - \left[\frac{1}{\gamma(k-1)} - 1 \right] \alpha \right) \right],$$

for $(x,y) \in \mathcal{X}$.

2.3.2 Special Cases of the M/Cox(r)/1 Queue

In this section we consider important special cases of the M/Cox(r)/1 queue. This includes queues with service-time distributions that are modeled by the hyper-exponential (H_r), the hypo-exponential (Hypo$_r$), the Erlang (E_r), and the exponential (M) distribution.

The $M/H_r/1$ Queue

The single server queue with hyper-exponentially distributed service times of order r is obtained by letting the service times consist only of one exponential phase with parameter μ_i with probability q_i for $i = 1, \ldots, r$. Note that the hyper-exponential distribution has the property that the coefficient of variation is greater than or equal to one. Unfortunately, the hyper-exponential distribution is not directly obtained from the Coxian distribution through interpretation, but rather from showing that the Laplace-transforms of the distribution functions are equal for specific parameter choices. In the case of $r = 2$ this result follows from, e.g., Appendix B of Tijms [24]. The general case is obtained by the following theorem.

Theorem 2.3 (Theorem 4 in [5]). *Under the assumption that* $\mu_1 > \cdots > \mu_r$, *a Coxian distribution with parameters* $(p_1, \ldots, p_{r-1}, \mu_1 \ldots, \mu_r)$ *is equivalent to a hyper-exponential distribution with parameters* $(q_1, \ldots, q_r, \mu_1, \ldots, \mu_r)$ *when the probabilities* p_i *are defined by*

$$p_i = \frac{\sum_{j=i+1}^{r} q_j \prod_{k=1}^{i} (\mu_k - \mu_j)}{\mu_i \sum_{j=i}^{r} q_j \prod_{k=1}^{i-1} (\mu_k - \mu_j)}, \tag{2.6}$$

for $i = 1, \ldots, r-1$.

The $M/$Hypo$_r/1$ Queue

The single server queue with hypo-exponentially distributed service times of order r is obtained by letting the service be the sum of r independent random variables that are exponentially distributed with parameter μ_i at phase i for $i = 1, \ldots, r$. Thus,

it can be obtained from the M/Cox(r)/1 queue by letting $p_1 = \cdots = p_{r-1} = 1$. The optimality equations are given by

$$\begin{aligned} g + \lambda V(0,0) &= \lambda V(1,0), \\ g + (\lambda + \mu_i)V(x,i-1) &= \lambda V(x+1,i-1) + \mu_i V(x,i) + x, \qquad i = 1,\ldots,r-1, \\ g + (\lambda + \mu_r)V(x,r-1) &= \lambda V(x+1,r-1) + \mu_r V(x-1,0) + x. \end{aligned}$$

Define $\beta(i) = \sum_{k=1}^{i}(1/\mu_k)$, then the average cost is given by

$$g = \frac{\lambda\beta(r)}{1-\lambda\beta(r)} - \frac{\lambda^2}{1-\lambda\beta(r)} \sum_{k=1}^{r} \frac{\beta(k-1)}{\mu_k},$$

and under the assumption $\lambda\beta(r) < 1$ the value function becomes

$$\begin{aligned} V(x,y) = \frac{\beta(r)x(x+1)}{2\big(1-\lambda\beta(r)\big)} - \frac{x}{1-\lambda\beta(r)} \left[\lambda \sum_{k=1}^{r} \frac{\beta(k-1)}{\mu_k} + \beta(y)\right] \\ + \frac{\lambda}{1-\lambda\beta(r)} \sum_{k=1}^{y} \frac{\beta(k-1)}{\mu_k}. \end{aligned}$$

The $M/E_r/1$ Queue

The single server queue with Erlang distributed service times of order r is obtained by letting the service be the sum of r independent random variables having a common exponential distribution. Thus, it can be obtained from the M/Cox(r)/1 queue by letting $p_1 = \cdots = p_{r-1} = 1$ and $\mu = \mu_1 = \cdots = \mu_r$. Note that the Erlang distribution can also be seen as a special case of the hypo-exponential distribution, and has a coefficient of variation equal to $1/r \leq 1$. The optimality equations are given by

$$\begin{aligned} g + \lambda V(0,0) &= \lambda V(1,0), \\ g + (\lambda + \mu)V(x,i-1) &= \lambda V(x+1,i-1) + \mu V(x,i) + x, \qquad i = 1,\ldots,r-1, \\ g + (\lambda + \mu)V(x,r-1) &= \lambda V(x+1,r-1) + \mu V(x-1,0) + x. \end{aligned}$$

The average cost is given by

$$g = \frac{\lambda r}{\mu - \lambda r} - \frac{\lambda^2 r(r-1)}{2\mu(\mu - \lambda r)},$$

and under the assumption $\lambda r/\mu < 1$ the value function becomes

$$V(x,y) = \frac{rx(x+1)}{2(\mu - \lambda r)} - \frac{x}{(\mu - \lambda r)} \left[\frac{\lambda r(r-1)}{2\mu} + y\right] + \frac{\lambda y(y-1)}{2\mu(\mu - \lambda r)}.$$

The M/M/1 Queue

The standard single server queue with exponentially distributed service times is obtained by having one phase in the M/Cox(r)/1 queue only, i.e., $r = 1$. Let $\mu = \mu_1$, then the optimality equations are equivalent to

$$g + (\lambda + \mu)V(x) = \lambda V(x+1) + \mu V([x-1]^+) + x,$$

where $[x]^+ = \max\{x, 0\}$. The average cost is given by $g = \lambda/(\mu - \lambda)$, and under the assumption $\lambda/\mu < 1$, the value function becomes

$$V(x) = \frac{x(x+1)}{2(\mu - \lambda)}.$$

2.3.3 The M/M/s Queue

Consider a queueing system with s identical independent servers. The arrivals are determined by a Poisson process with parameter λ. The service times are exponentially distributed with parameter μ. An arriving customer that finds no idle server waits in a buffer of infinite size. We focus on the average number of customers in the system represented by generating a cost of x monetary units when x customers are present in the system. The Poisson equations for this M/M/s queue are then given by

$$\begin{aligned}
g + \lambda V(0) &= \lambda V(1), \\
g + (\lambda + x\mu)V(x) &= x + \lambda V(x+1) + x\mu V(x-1), && x = 1, \ldots, s-1, \\
g + (\lambda + s\mu)V(x) &= x + \lambda V(x+1) + s\mu V(x-1), && x = s, s+1, \ldots,
\end{aligned}$$

From the first equation we can deduce that $V(1) = g/\lambda$. The second equation can be written as Eq. (2.3) with $\alpha(x) = -(\lambda + x\mu)/\lambda$, $\beta(x) = x\mu/\lambda$, and $q(x) = g/\lambda$. From Theorem 2.1 we have that

$$V(x) = \frac{g}{\lambda} \sum_{i=1}^{x} \sum_{k=0}^{i-1} \frac{(i-1)!}{(i-k-1)!} \left(\frac{\lambda}{\mu}\right)^{-k} - \frac{1}{\lambda} \sum_{i=1}^{x} (i-1) \sum_{k=0}^{i-2} \frac{(i-2)!}{(i-k-2)!} \left(\frac{\lambda}{\mu}\right)^{-k}.$$

Let $\rho = \lambda/(s\mu)$. For $x = s, s+1, \ldots$ we have

$$V(x) = V(s) - \frac{(x-s)\rho}{1-\rho} \frac{g}{\lambda} + \left[\frac{(x-s)(x-s+1)\rho}{2(1-\rho)} + \frac{(x-s)\big(\rho + s(1-\rho)\big)\rho}{(1-\rho)^2} \right] \frac{1}{\lambda} +$$

$$\frac{(1/\rho)^{x-s} - 1}{1-\rho} \left[\frac{\rho}{1-\rho} \frac{g}{\lambda} + h(s) - h(s-1) - \frac{\big(\rho + s(1-\rho)\big)\rho}{\lambda(1-\rho)^2} \right].$$

The long-term expected average costs, which represents the average number of customers in the system in this case, is given by

$$g = \frac{(s\rho)^s \rho}{s!(1-\rho)^2}\left[\sum_{n=0}^{s-1}\frac{(s\rho)^n}{n!} + \frac{(s\rho)^s}{s!(1-\rho)}\right]^{-1} + s\rho.$$

2.3.4 The Blocking Costs in an M/M/s/s Queue

Consider the previous queueing system with s identical independent servers but with no buffers for customers to wait. An arriving customer that finds no idle server is blocked and generates a cost of one monetary unit. Let state x denote the number of customers in the system. The Poisson equations for this M/M/s/s queue are then given by

$$\begin{aligned} g + \lambda V(0) &= \lambda V(1), \\ g + (\lambda + x\mu)V(x) &= \lambda V(x+1) + x\mu V(x-1), \qquad x = 1,\ldots,s-1, \\ g + s\mu V(s) &= \lambda + s\mu V(s-1). \end{aligned}$$

In order to obtain the relative value function V, we repeat the argument as in the case of the M/M/s queue. This yields the following value function.

$$\begin{aligned} V(x) &= \frac{g}{\lambda}\sum_{i=1}^{x}\sum_{k=0}^{i-1}\frac{(i-1)!}{(i-k-1)!}\left(\frac{\lambda}{\mu}\right)^{-k} \\ &= \frac{(\lambda/\mu)^s/s!}{\sum_{i=0}^{s}(\lambda/\mu)^i/i!}\sum_{i=1}^{x}\sum_{k=0}^{i-1}\frac{(i-1)!}{(i-k-1)!}\left(\frac{\lambda}{\mu}\right)^{-k}. \end{aligned} \tag{2.7}$$

The average costs is given by

$$g = \frac{(\lambda/\mu)^s/s!}{\sum_{i=0}^{s}(\lambda/\mu)^i/i!}\,\lambda.$$

2.3.5 Priority Queues

Consider a queueing system with two classes of customers arriving according to a Poisson process. There is only one server available serving either a class-1 or a class-2 customer with exponentially distributed service times. A class-i customer has arrival rate λ_i, and is served with rate μ_i for $i = 1,2$. The system is subject to holding costs and switching costs. The cost of holding a class-i customer in the system for one unit of time is c_i for $i = 1,2$. The cost of switching from serving a class-1 to a class-2 customer (from a class-2 to a class-1 customer) is s_1 (s_2, respectively).

The system follows a priority discipline indicating that class-1 customers have priority over class-2 customers. The priority is also preemptive, i.e., when serving a class-2 customer, the server switches immediately to serve a class-1 customer upon arrival of a class-1 customer. Upon emptying the queue of class-1 customers, the service of class-2 customers, if any present, is resumed from the point where it was interrupted. Due to the exponential service times, this is equivalent to restarting the service for this customer.

Let state (x,y,z) for $x,y \in \mathbb{N}_0$, $z \in \{1,2\}$ denote that there are x class-1 and y class-2 customers present in the system, with the server serving a class-z customer, if present. Let $\rho_i = \lambda_i/\mu_i$ for $i = 1,2$ and assume that the stability condition $\rho_1 + \rho_2 < 1$ holds. Then the Markov chain is stable and $g < \infty$ holds. Furthermore, the Markov chain satisfies the unichain condition, hence the Poisson equations are given by

$$\begin{aligned}
g + (\lambda_1+\lambda_2+\mu_1)V(x,y,1) &= c_1 x + c_2 y + \lambda_1 V(x+1,y,1) + \lambda_2 V(x,y+1,1) + \\
&\quad \mu_1 V(x-1,y,1), && x > 0, y \geq 0, \\
V(0,y,1) &= s_1 + V(0,y,2), && y > 0, \\
V(x,y,2) &= s_2 + V(x,y,1), && x > 0, y \geq 0, \\
g + (\lambda_1+\lambda_2+\mu_2)V(0,y,2) &= c_2 y + \lambda_1 V(1,y,2) + \lambda_2 V(0,y+1,2) + \\
&\quad \mu_2 V(0,y-1,2), && y > 0, \\
g + (\lambda_1+\lambda_2)V(0,0,z) &= \lambda_1 V(1,0,z) + \lambda_2 V(0,1,z), && z = 1,2.
\end{aligned}$$

First observe that, given the values of $V(x,y,1)$, the values of $V(x,y,2)$ are easily obtained by considering the difference equations: $V(x,y,2) = V(x,y,1) + (\lambda_1 s_2 - \lambda_2 s_1)/(\lambda_1+\lambda_2)\mathbb{1}(x=0,y=0) - s_1\mathbb{1}(x=0,y>0) + s_2\mathbb{1}(x>0,y\geq 0)$. Therefore, we will only show the expression for $V(x,y,1)$, as $V(x,y,2)$ is easily derived from it. Let $V = V_g + V_{c_1} + V_{c_2} + V_{s_1} + V_{s_2}$ be the decomposition of the value function. Then the previous observation directly prescribes that V_g, V_{c_1}, and V_{c_2} are independent of z. Furthermore, the value function V_{c_1} equals the value function of the single server queue due to the preemptive priority behavior of the system.

The other value functions can be derived by setting $V(x,y) = c_1 x^2 + c_2 x + c_3 y^2 + c_4 y + c_5 xy + c_6$ with constants c_i for $i = 1,\ldots,6$ to be determined. Substitution of this equality into the optimality equations yields a new set of equations that is easier to solve. Let θ be the unique root $\theta \in (0,1)$ of the equation

$$\lambda_1 x^2 - (\lambda_1+\lambda_2+\mu_1)x + \mu_1 = 0.$$

Then, solving for the constants yields the solution to the optimality equations, which is given by

$$V_g(x,y,z) = -\frac{x}{\mu_1(1-\rho_1)}\,g,$$

$$V_{c_1}(x,y,z) = \frac{x(x+1)}{2\mu_1(1-\rho_1)}\,c_1 + \frac{\rho_1 x}{\mu_1(1-\rho_1)^2}\,c_1,$$

$$V_{c_2}(x,y,z) = \frac{\lambda_2 x(x+1)}{2\mu_1^2(1-\rho_1)(1-\rho_1-\rho_2)}c_2 + \frac{\rho_2(\mu_1-\lambda_1+\mu_2\rho_1)x}{\mu_1^2(1-\rho_1)^2(1-\rho_1-\rho_2)}c_2 +$$

$$\frac{y(y+1)}{2\mu_2(1-\rho_1-\rho_2)}c_2 + \frac{xy}{\mu_1(1-\rho_1-\rho_2)}c_2,$$

$$V_{s_i}(x,y,1) = \frac{\lambda_1\theta}{\lambda_1+\lambda_2}\frac{\rho_1\rho_2 x}{1-\rho_1}s_i + \frac{\lambda_1\theta}{\lambda_1+\lambda_2}\frac{\lambda_1 y}{\mu_2}s_i + \frac{\lambda_1}{\lambda_1+\lambda_2}s_i\mathbb{1}(y>0) +$$

$$\frac{\lambda_1}{\lambda_1+\lambda_2}(1-\theta^x)s_i\mathbb{1}(x>0,y=0), \qquad i=1,2,$$

with

$$g = \frac{\rho_1}{1-\rho_1}c_1 + \frac{\rho_2(\mu_1-\mu_1\rho_1+\mu_2\rho_1)}{\mu_1(1-\rho_1)(1-\rho_1-\rho_2)}c_2 + (s_1+s_2)\times$$

$$\left[\lambda_1\left\{\rho_1\left(\frac{\lambda_1\theta}{\lambda+\lambda_2}-1\right)+\frac{\lambda_1}{\lambda_1+\lambda_2}(1-\theta)\right\}+\lambda_2\left\{\frac{\lambda_1\theta}{\lambda_1+\lambda_2}\frac{\lambda_1}{\mu_2}+\frac{\lambda_1}{\lambda_1+\lambda_2}\right\}\right].$$

2.4 Application: Routing to Parallel Queues

In this section we illustrate how the value function can be used to obtain nearly optimal policies for dynamic routing problems to parallel single server queues. The general idea is to start with a policy such that each queue behaves as an independent single server queue. By doing so, the average cost and the value function can be readily determined from the results of the previous sections. Next, one step of policy iteration is performed to obtain an improved policy without having to compute the policy iteratively.

The control problem that we study can be formalized as follows. Consider two parallel infinite buffer single server queues. The service times of server i are Coxian distributed with order r_i with parameters $(p_1^i,\ldots,p_{r_i-1}^i,\mu_1^i,\ldots,\mu_{r_i}^i)$ for $i=1,2$. Furthermore, queue i has holding costs h_i for $i=1,2$. An arriving customer can be sent to either queue 1 or queue 2. The objective is to minimize the average costs. Note that the distribution of the service times does not necessarily need to be Coxian. Since the class of Coxian distributions is dense in the set of non-negative distribution functions, we can approximate these distributions with a Coxian distribution by using, e.g., the Expectation-Maximization (EM) algorithm (see [2]). The EM algorithm is an iterative scheme that minimizes the information divergence given by the Kullback-Leibler information for a fixed order r. For the cases we have consid-

ered, a Coxian distribution of order $r = 5$ was sufficiently adequate to describe the original distribution.

Let x_i be the number of customers in queue i, and y_i the number of completed phases of the customer in service, if there is one, at queue i for $i = 1,2$. Given that the service time distribution is adequately described by a Coxian distribution, the optimality equations for this system are given by

$$\begin{aligned} g + \big(\lambda + \mathbb{1}_{\{x_1>0\}}\,\mu^1_{y_1+1} + \mathbb{1}_{\{x_2>0\}}\,\mu^2_{y_2+1}\big)V(x_1,y_1,x_2,y_2) &= h_1x_1 + h_2x_2 + \\ \lambda \min\{V(x_1+1,x_2,y_1,y_2), V(x_1,x_2+1,y_1,y_2)\} &+ \\ \mathbb{1}_{\{x_1>0\}}\,\mu^1_{y_1+1}[p^1_{y_1+1}V(x_1,x_2,y_1+1,y_2) + \overline{p}^1_{y_1+1}V(x_1-1,x_2,0,y_2)] &+ \\ \mathbb{1}_{\{x_2>0\}}\,\mu^2_{y_2+1}[p^2_{y_2+1}V(x_1,x_2,y_1,y_2+1) + \overline{p}^2_{y_2+1}V(x_1,x_2-1,y_1,0)], & \end{aligned}$$

for $(x_i,y_i) \in \{(0,0)\} \cup \mathbb{N} \times \{0,\ldots,r_i-1\}$ with $p^i_{r_i} = 0$ and $\overline{p}^i_y = 1 - p^i_y$ when $i = 1,2$.

Take as initial control policy the Bernoulli policy with parameter $\eta \in [0,1]$, i.e., the policy that splits the arrivals into two streams such that arrivals occur with rate $\eta\lambda$ at queue 1, and with rate $(1-\eta)\lambda$ at queue 2. Under the Bernoulli policy, the optimality equation is obtained by replacing the term $\lambda \min\{V(x_1+1,x_2,y_1,y_2), V(x_1,x_2+1,y_1,y_2)\}$ with $\eta\lambda V(x_1+1,x_2,y_1,y_2) + (1-\eta)\lambda V(x_1,x_2+1,y_1,y_2)$. Hence, it follows that the two queues behave as independent single server queues for which the average cost g_i and the value function V_i for $i = 1,2$ can be determined from Theorem 2.2. The average cost g' and the value function V' for the whole system is then given by the sum of the individual average costs $g = g_1 + g_2$ and the sum of the individual value functions $V' = V_1 + V_2$, respectively. Note that for a specified set of parameters, the optimal Bernoulli policy obtained by minimizing with respect to η is straightforward. We shall therefore use the optimal Bernoulli policy in the numerical examples. The policy improvement step now follows from the minimizing action in $\min\{V'(x_1+1,x_2,y_1,y_2), V'(x_1,x_2+1,y_1,y_2)\}$.

The Coxian distribution allows for many interesting numerical experiments. Therefore, we restrict ourselves to four representative examples that display the main ideas. We shall use the notation g_B, g', and g^* for the average cost obtained under the optimal Bernoulli policy, the one-step improved policy, and the optimal policy, respectively. Moreover, we set $h_1 = h_2 = 1$, and $\lambda = \frac{3}{2}$ for the first example, and $\lambda = 1$ for the other three examples.

We first start with two queues having a Coxian $C(2)$ distribution. Queue 1 has an Erlang E_2 distribution with parameter $\mu = 2$, such that the mean and the variance of the service time is 1 and 2, respectively. The parameters at queue 2 are chosen such that the mean remains 1, but the variance increases to 3, 4, and 5, respectively. Table 2.1 summarizes the results, and shows that the one-step improved policy has a close to optimal value. The proportional extra cost $(g' - g^*)/g^*$ is practically zero in all cases.

Parameters for queue 2	g_B	g'	g^*
$p^2 = \frac{2}{3}, \mu^2 = (2, \frac{4}{3})$	5.147786	3.208688	3.208588
$p^2 = \frac{1}{2}, \mu^2 = (2, 1)$	5.405949	3.332179	3.332038
$p^2 = \frac{2}{5}, \mu^2 = (2, \frac{4}{5})$	5.652162	3.445815	3.445787

Table 2.1: Numerical results for $r = 2$ with $p^1 = 1, \mu^1 = (2,2)$

Next, we show a similar experiment with $r = 5$. The service distribution at queue 1 is fixed at a hypo-exponential Hypo_5 distribution with parameter $\mu = (2,3,2,3,4)$. Again the one-step improved policy performs quite well. Table 2.2 shows that the greatest proportional extra cost is given by 0.03 (the third experiment).

Parameters for queue 2	g_B	g'	g^*
$p^2 = (\frac{9}{10}, \frac{4}{5}, \frac{7}{10}, \frac{3}{5}), \mu^2 = (2,3,2,3,4)$	6.175842	3.787954	3.783727
$p^2 = (\frac{3}{5}, \frac{7}{10}, \frac{4}{5}, \frac{9}{10}), \mu^2 = (2,3,2,3,4)$	3.729859	2.493349	2.480818
$p^2 = (\frac{2}{5}, \frac{1}{5}, \frac{4}{5}, \frac{1}{2}), \quad \mu^2 = (3,2,4,2,3)$	1.399628	1.169286	1.132408

Table 2.2: Numerical results for $r = 5$ with $p^1 = (1,1,1,1), \mu^1 = (2,3,2,3,4)$

In the third example we take an Erlang E_2 distribution with parameter $\mu = 2$ at queue 1, and a lognormal distribution with parameters $\mu = 0.5$ and $\sigma = 1$ at queue 2. Recall that the probability density function f of the lognormal distribution is given by

$$f(x) = \frac{1}{x\sqrt{2\pi}\sigma} \cdot e^{-\frac{1}{2}\left(\frac{\ln(x)-\mu}{\sigma}\right)^2},$$

for $x > 0$. We approximate the lognormal distribution with a $\text{Cox}(r)$ distribution of order $r = 2, 5, 10$, and 20, respectively, using the EM-algorithm. We also compare this with a two-moment fit of the Coxian distribution. Let X be a random variable having a coefficient of variation $c_X \geq \frac{1}{2}\sqrt{2}$, then the following is suggested by Marie [13]: $\mu_1 = 2/\mathbb{E}X$, $p_1 = 0.5/c_X^2$, and $\mu_2 = p_1\mu_1$.

The results of the EM-algorithm and the 2-moment fit are displayed in Fig. 2.1. The fit with the Cox(20) distribution is omitted, since it could not be distinguished from the lognormal probability density. Therefore, the optimal value when using the Cox(20) distribution can be considered representative for the optimal value when using the lognormal distribution. Note that the EM-approximation with the Cox(2) distribution captures more characteristics of the lognormal distribution than the 2-moment fit. This result is also reflected in Table 2.3, since the value of the optimal policy g^* for the Cox(2) distribution is closer to the value of g^* when the Cox(20) distribution is used. The greatest proportional extra cost for the EM-approximations is given by 0.02 (EM fit with $r = 2$).

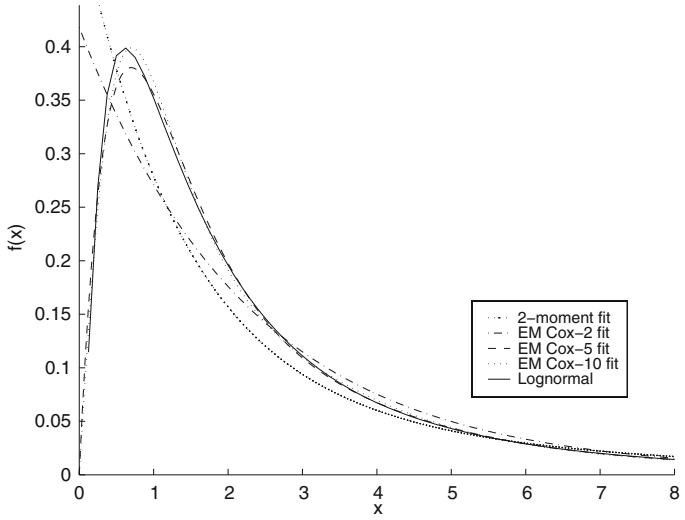


Fig. 2.1: Lognormal ($\mu = 0.5$, $\sigma = 1$) probability density

Approximation for queue 2	g_B	g'	g^*
2-moment fit	4.617707	3.021571	2.976950
EM algorithm with $r = 2$	4.554838	2.982955	2.933345
EM algorithm with $r = 5$	4.526013	2.965625	2.919847
EM algorithm with $r = 10$	4.527392	2.963318	2.917011
EM algorithm with $r = 20$	4.527040	2.963311	2.917169

Table 2.3: Lognormal distribution: numerical results with $p^1 = 1, \mu^1 = (2,2)$

In the final example we take an Erlang E_2 distribution with parameter $\mu = 2$ at queue 1, and a Weibull distribution with parameters $a = 0.3$ and $b = 0.107985$. Recall that the probability density function f of the Weibull distribution is given by

$$f(x) = ax^{a-1} \frac{e^{-(\frac{x}{b})^a}}{b^a},$$

for $x > 0$. Note that the parameters a and b are chosen such that the Weibull distribution has mean 1. We approximate the Weibull distribution by a Cox(r) distribution of order $r = 2, 5, 10$, and 20, respectively, using the EM-algorithm. The results of the EM-algorithm are depicted in Fig. 2.2. Again we omitted the fit of the Cox(20) distribution, since it could not be distinguished from the Weibull probability density. Moreover, since the coefficient of variation is less than $\frac{1}{2}\sqrt{2}$ we also omitted the two-moment fit. The results in Table 2.4 again indicate that the one-step improved policy has a close to optimal value, since the proportional extra cost is practically zero.

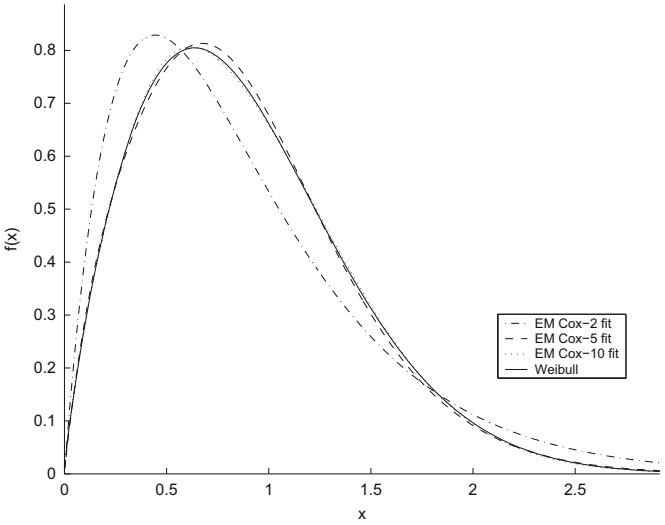


Fig. 2.2: Weibull ($a = 1.8$, $b = 1$) probability density

Approximation for queue 2	g_B	g'	g^*
EM algorithm with $r = 2$	1.563547	1.167233	1.167232
EM algorithm with $r = 5$	1.524032	1.148638	1.148511
EM algorithm with $r = 10$	1.522566	1.148570	1.148100
EM algorithm with $r = 20$	1.522387	1.148826	1.148552

Table 2.4: Weibull distribution: numerical results with $p^1 = 1, \mu^1 = (2,2)$

The previous examples show that the one-step policy improvement method yields nearly optimal policies, even when non-Markovian service distributions are approximated by a Coxian distribution. For the lognormal and the Weibull distribution that we studied, a Coxian distribution of order $r = 5$ was already sufficient for an adequate approximation. Note that the one-step improved policy can be easily obtained for more than two queues. In this section we restricted attention to two queues, since the numerical computation of the value of the optimal policy g^* becomes rapidly intractable for more than two queues. Observe that the computational complexity is exponential in the number of queues in contrast to a single step of policy iteration that has linear complexity in the number of queues.

2.5 Application: Dynamic Routing in Multiskill Call Centers

Consider a multi-skill call center in which agents handle calls that require different skills. We represent the skill set $\mathcal{S}$ by $\mathcal{S} = \{1, \ldots, N\}$. We assume that calls that require skill $s \in \mathcal{S}$ arrive to the call center according to a Poisson process with rate λ_s. Let $\mathcal{G} = \mathcal{P}(\mathcal{S})$ denote the groups with different skill sets defined by the power set of all the skills. Let $\mathcal{G}_s = \{G \in \mathcal{G} \,|\, s \in G\}$ denote all skill groups that include skill $s \in \mathcal{S}$. The agents are grouped according to their skills and can therefore be indexed by the elements of $\mathcal{G}$. Each group $G \in \mathcal{G}$ consists of S_G agents, and serves a call that requires a skill within group G with independent exponentially distributed times with parameter μ_G.

Scenario 1: A Call Center with No Waiting Room

Suppose that we are dealing with a loss system, i.e., there are no buffers at all in our model. Hence, there is no common queue for calls to wait, so every arriving call has either to be routed to one of the agent groups immediately, or has to be blocked and is lost. Also, when a call is routed to a group that has no idle servers left, the call is lost. The objective in this system is to minimize the average number of lost calls.

Let us formulate the problem as a Markov decision problem. Fix an order of the elements of $\mathcal{G}$, say $\mathcal{G} = (G_1, \ldots, G_{2^N-1})$, and define the state space of the Markov decision problem by $\mathcal{X} = \prod_{i=1}^{2^N-1} \{0, \ldots, S_{G_i}\}$. Then the $|\mathcal{G}|$-dimensional vector $\mathbf{x} \in \mathcal{X}$ denotes the state of the system according to the fixed order, i.e., x_G is the number of calls in service at group $G \in \mathcal{G}$. Also, represent by the $|\mathcal{G}|$-dimensional unit vector e_G the vector with zero at every entry, except for a one at the entry corresponding to G according to the fixed order. When a call that requires skill $s \in \mathcal{S}$ arrives, the decision maker can choose to block the call, or to route the call to any agent group $G \in \mathcal{G}_s$ for which $x_G < S_G$. Hence, the action set is defined by $\mathcal{A}_s = \{G \in \mathcal{G}_s \,|\, x_G < S_G\} \cup \{b\}$, where action b stands for blocking the call. Therefore, for an arriving call that requires skill $s \in \mathcal{S}$ the transition rates $p(\mathbf{x}, a, \mathbf{y})$ of going from $\mathbf{x} \in \mathcal{X}$ to $\mathbf{y} \in \mathcal{X}$ after taking decision $a \in \mathcal{A}_s$ are given by $p(\mathbf{x}, b, \mathbf{x}) = \lambda_s$, $p(\mathbf{x}, a, \mathbf{x} + e_a) = \lambda_s$ when $a \neq b$, and zero otherwise. The transition rates for the service completions are given by $p(\mathbf{x}, a, \mathbf{x} - e_G) = x_G \mu_G$ for $\mathbf{x} \in \mathcal{X}$, $G \in \mathcal{G}$ and any action a. The objective is modeled by the cost function $c(\mathbf{x}, a) = 1$ for any $\mathbf{x} \in \mathcal{X}$ and $a = b$.

The tuple $(\mathcal{X}, \{\mathcal{A}_s\}_{s \in \mathcal{S}}, p, c)$ defines the Markov decision problem. After uniformizing the system (see Sect. 11.5 of Puterman [18]) we obtain the dynamic programming optimality equation for the system given by

$$g + \Big[\sum_{s\in\mathcal{S}} \lambda_s + \sum_{G\in\mathcal{G}} S_G\mu_G\Big] V(\mathbf{x}) = \sum_{s\in\mathcal{S}} \lambda_s \min\{1 + V(\mathbf{x}), V(\mathbf{x}+e_G) \,|\, G \in \mathcal{G}_s, x_G < S_G\}$$
$$+ \sum_{G\in\mathcal{G}} x_G\mu_G V(\mathbf{x}-e_G) + \sum_{G\in\mathcal{G}} (S_G - x_G)\mu_G V(\mathbf{x}), \tag{2.8}$$

where g is the long-term expected average costs, and V is the relative value function.

Note the unusual place of the minimization in Expression (2.8): the actions in our case depend on the type of arriving call. It is easy to rewrite the optimality equation in standard formulation (see, e.g., Chap. 5 of Koole [11]), but this would complicate the notation considerably.

For the problem of skill-based routing we will base the approximate relative value function on a static randomized policy for the initial routing policy. Let $\mathcal{G}^{(n)} = \{G \in \mathcal{G} \,|\, |G| = n\}$ be all agent groups that have exactly n skills for $n = 1,\ldots,N$. Define accordingly, $\mathcal{G}_s^{(n)} = \{G \in \mathcal{G}^{(n)} \,|\, s \in G\}$ to be all agent groups that have n skills, including skill $s \in \mathcal{S}$. For a given skill s, this creates a hierarchical structure $\mathcal{G}_s^{(1)},\ldots,\mathcal{G}_s^{(N)}$ from specialized agents having only one skill to cross-trained generalists having all skills. For a call center with three skills, we would have three levels in the hierarchy: the specialists (groups $\{1\}$, $\{2\}$, and $\{3\}$), the agents with two skills (groups $\{1, 2\}$, $\{1, 3\}$, and $\{2, 3\}$), and the generalists (group $\{1, 2, 3\}$). In the following paragraphs we will describe the steps in the one-step policy improvement algorithm in more detail using this call center with three skills as illustration.

Initial Policy

The initial policy $\hat{\pi}$, on which we will base the approximate relative value function, tries to route a call requiring skill s through the hierarchy, i.e., it tries $\mathcal{G}_s^{(1)}$ first, and moves to $\mathcal{G}_s^{(2)}$ if there are no idle agents in $\mathcal{G}_s^{(1)}$. In $\mathcal{G}_s^{(2)}$ there may be more than one group that one can route to. The initial policy routes the call according to fixed probabilities to the groups in $\mathcal{G}_s^{(2)}$, i.e., it splits the overflow stream from $\mathcal{G}_s^{(1)}$ into fixed fractions over the groups in $\mathcal{G}_s^{(2)}$. The call progresses through the hierarchy whenever it is routed to a group with no idle agents until it is served at one of the groups, or it is blocked at $\mathcal{G}_s^{(N)}$ eventually. The rationale behind the policy that sends calls to the agents with the fewest number of skills first has been rigorously justified in [9, 16]. To illustrate this for three skills: the overflow of calls from group $\{1\}$ are split by fixed fractions α and $\overline{\alpha} = 1 - \alpha$ in order to be routed to groups $\{1,2\}$ and $\{1,3\}$, respectively. The overflow process from groups $\{2\}$ and $\{3\}$ are treated accordingly by the fractions β and γ, respectively.

We have yet to define how to choose the splitting probabilities in the initial routing policy. In order to define these, we ignore the fact that the overflow process is not a Poisson process, and we consider all overflows to be independent. Thus, we do

as if the overflow process at group $G \in \mathcal{G}_s^{(i)}$ is a Poisson process with rate λ_G times the blocking probability. Together with the splitting probabilities, one can compute the rate of the Poisson arrivals at each station in $\mathcal{G}^{(i+1)}$ composed of the assumed independent Poisson processes. The splitting probabilities are then chosen such that the load at every group in $\mathcal{G}^{(i+1)}$ is balanced. To illustrate this for the call center with three skills, recall the Erlang loss formula $B(\lambda,\mu,S)$ for an M/M/*S*/*S* queue with arrival rate λ and service rate μ,

$$B(\lambda,\mu,S) = \frac{(\lambda/\mu)^S/S!}{\sum_{i=0}^{S}(\lambda/\mu)^i/i!}. \tag{2.9}$$

The overflow rate at group $\{i\}$ for $i = 1,2,3$ is then given by $\lambda_i B(\lambda_i,\mu_{\{i\}},S_{\{i\}})$. Hence, the arrival rate at the groups in $\mathcal{G}^{(2)}$ are given by

$$\begin{aligned}
\lambda_{\{1,2\}} &= \lambda_1 B(\lambda_1,\mu_{\{1\}},S_{\{1\}})\,\alpha + \lambda_2 B(\lambda_2,\mu_{\{2\}},S_{\{2\}})\,\beta,\\
\lambda_{\{1,3\}} &= \lambda_1 B(\lambda_1,\mu_{\{1\}},S_{\{1\}})\,(1-\alpha) + \lambda_3 B(\lambda_3,\mu_{\{3\}},S_{\{3\}})\,\gamma,\\
\lambda_{\{2,3\}} &= \lambda_2 B(\lambda_2,\mu_{\{2\}},S_{\{2\}})\,(1-\beta) + \lambda_3 B(\lambda_3,\mu_{\{3\}},S_{\{3\}})\,(1-\gamma).
\end{aligned}$$

The splitting probabilities can be determined by minimizing the average cost in the system. Since the average cost under this policy is the sum of the blocking probabilities of each queue in the system, the optimal splitting probability could create asymmetry in the system, i.e., one queue has a very low blocking probability whereas another has a high blocking probability. Numerical experiments show that a situation with a more balanced blocking probability over all queues leads to better one-step improved policies. Therefore, we choose the splitting probabilities such that

$$\frac{\lambda_{\{1,2\}}}{S_{\{1,2\}}\mu_{\{1,2\}}} = \frac{\lambda_{\{1,3\}}}{S_{\{1,3\}}\mu_{\{1,3\}}} = \frac{\lambda_{\{2,3\}}}{S_{\{2,3\}}\mu_{\{2,3\}}}.$$

In case this equation does not have a feasible solution, we choose the splitting probabilities such that we minimize

$$\left|\frac{\lambda_{\{1,2\}}}{S_{\{1,2\}}\mu_{\{1,2\}}} - \frac{\lambda_{\{1,3\}}}{S_{\{1,3\}}\mu_{\{1,3\}}}\right| + \left|\frac{\lambda_{\{1,2\}}}{S_{\{1,2\}}\mu_{\{1,2\}}} - \frac{\lambda_{\{2,3\}}}{S_{\{2,3\}}\mu_{\{2,3\}}}\right| + \left|\frac{\lambda_{\{1,3\}}}{S_{\{1,3\}}\mu_{\{1,3\}}} - \frac{\lambda_{\{2,3\}}}{S_{\{2,3\}}\mu_{\{2,3\}}}\right|.$$

Finally, the arrival rate at the last group with all skills is given by

$$\lambda_{\{1,2,3\}} = \sum_{G\in\mathcal{G}^{(2)}} \lambda_G B(\lambda_G,\mu_G,S_G).$$

Policy Evaluation

In the policy evaluation step, one is required to solve the long-run expected average costs and the relative value function from the Poisson equations for the policy $\hat{\pi}$. Note that for our purpose of deriving an improved policy, it suffices to have the relative value function only. Under the initial policy this leads to solving the relative value function of a series of multi-server queues in tandem. This is a difficult problem that is yet unsolved even for tandem series of single server queues. Therefore, we choose to approximate the relative value function based on the assumptions made in defining the initial policy.

For the approximation we assume that the overflow process is indeed a Poison process and that they are independent from all other overflow processes. We approximate the relative value function by the sum of the relative value functions for each agent group. More formally, let $V_G(x_G, \lambda_G, \mu_G, S_G)$ be the relative value function for an M/M/S_G/S_G system with arrival rate λ_G and service rate μ_G, evaluated when there are x_G calls present. The value function for the whole system is then approximated by

$$\hat{V}(\mathbf{x}) = \sum_{G \in \mathcal{G}} V_G(x_G, \lambda_G, \mu_G, S_G). \tag{2.10}$$

Policy Improvement

By substitution of the relative value function determined in the policy evaluation step into the optimality equations, one can derive a better policy by evaluating

$$\min\{1 + \hat{V}(\mathbf{x}), \hat{V}(\mathbf{x} + e_G) \,|\, G \in \mathcal{G}_s, x_G < S_G\} \; =$$

$$\min\{1, V_G(x_G + 1, \lambda_G, \mu_G, S_G) - V_G(x_G, \lambda_G, \mu_G, S_G) \,|\, G \in \mathcal{G}_s, x_G < S_G\}.$$

The last term follows from subtracting $V(\mathbf{x})$ from all terms in the minimization and using the linear structure of $V(\mathbf{x})$.

In Table 2.5 we find the results of the one-step policy improvement algorithm for scenario 1. The parameters for λ_X, μ_X, and S_X in the table are organized for the groups $\{1\}$, $\{2\}$, and $\{3\}$. The parameters for μ_{XY} and S_{XY} are ordered for groups $\{1,2\}$, $\{1,3\}$, and $\{2,3\}$, respectively. The greatest proportional extra cost $\Delta = (g' - g^*)/g^*$ over all experiments in Table 2.5 is 12.9% (the last experiment). Other experiments show that the proportional extra costs lies within the range of 0–13%. Thus, we can see that the improved policy has a good performance already after one step of policy improvement.

Scenario 2: A Call Center with Specialists and Generalists Only

Suppose that we are dealing with a call center having agents with one skill (specialists) and fully cross-trained agents (generalists) only, i.e., $\mathcal{G} = (G_1, \ldots, G_{|\mathcal{S}|}, G_{|\mathcal{S}|+1}) = (\{1\}, \ldots, \{N\}, \{1, \ldots, N\})$. In this scenario we assume that calls that require skill $s \in \mathcal{S}$ are pooled in a common infinite buffer queue after which they are assigned to

λ_X	μ_X	μ_{XY}	S_X	S_{XY}	g	g'	g^*	Δ
6 6 6	1.0 1.0 1.0	1.0 1.0 1.0	2 2 2	2 2 2	0.361	0.349	0.344	1.505
6 5 4	2.0 1.5 1.0	1.5 1.0 1.0	2 2 2	2 2 2	0.170	0.147	0.143	2.781
7 6 5	2.0 1.5 1.0	1.5 1.5 1.0	3 3 3	2 2 2	0.119	0.099	0.096	3.264
7 6 5	1.5 1.0 1.0	1.5 1.0 1.0	3 3 3	3 3 2	0.130	0.107	0.103	3.930
6 5 4	2.0 1.5 1.0	1.5 1.0 1.0	3 3 3	2 2 2	0.072	0.057	0.054	5.672
6 5 4	2.0 1.5 1.0	1.5 1.5 1.0	3 3 3	2 2 2	0.061	0.046	0.042	8.011
10 4 4	1.5 1.0 1.0	1.5 1.5 1.0	2 2 2	2 2 2	0.281	0.274	0.252	8.816
10 6 3	1.5 1.0 1.0	1.5 1.0 1.0	3 3 3	3 3 2	0.160	0.148	0.131	12.851

Table 2.5: Numerical results for scenario 1 with $\mu_{\{1,2,3\}} = 1$ and $S_{\{1,2,3\}} = 2$

a specialist or a generalist in a first-come first-served order. The objective in this system is to minimize the average number of calls in the system, which in its turn is related to the average waiting time of a call.

In addition to the state of the agents groups, we also have to include the state of the queues in the state space. For this purpose, let the $|\mathcal{S}|$-dimensional vector $\mathbf{q}$ denote the state of the queues, i.e., q_s is number of calls in queue s that require skill $s \in \mathcal{S}$. Moreover, the system also has to address the agent-selection problem and the call-selection problem. The first problem occurs when one has to assign an agent to an arriving call. The second problem occurs when an agent becomes idle and a potential call in the queue can be assigned. Following the same approach as in scenario 1, we obtain the dynamic programming optimality equation given by

$$\begin{aligned} g + \Big[\sum_{s\in\mathcal{S}} \lambda_s + \sum_{G\in\mathcal{G}} S_G \mu_G\Big] V(\mathbf{q},\mathbf{x}) = \sum_{s\in\mathcal{S}} q_s + \sum_{G\in\mathcal{G}} x_G + \sum_{s\in\mathcal{S}} \lambda_s \tilde{V}(\mathbf{q}+e_s,\mathbf{x}) + \\ \sum_{G\in\mathcal{G}} x_G \mu_G \tilde{V}(\mathbf{q},\mathbf{x}-e_G) + \sum_{G\in\mathcal{G}} (S_G - x_G)\mu_G V(\mathbf{q},\mathbf{x}), \end{aligned} \tag{2.11}$$

where $\tilde{V}(\mathbf{q},\mathbf{x}) = \min\{V(\mathbf{q}-e_s,\mathbf{x}+e_G) \,|\, s \in \mathcal{S}, G \in \mathcal{G}_s, q_s > 0, x_G < S_G\} \cup \{V(\mathbf{q},\mathbf{x})\}$. The first two terms on the right-hand side represent the cost structure, i.e., the number of calls in the system. The third and fourth term represent the agent-selection and the call-selection decisions, respectively.

Initial Policy

In principle, we could take as initial policy $\hat{\pi}$ a similar policy as used in scenario 1: a fraction α_s of type s calls are assigned to the corresponding specialists, and a fraction $1-\alpha_s$ are assigned to the generalists. Instead of using the relative value function for the M/M/S/S queue, we could use the relative value function for the M/M/S queue. However, this approximation would then assume a dedicated queue in front of the group of generalists. Consequently, a customer that is sent to the group of generalists that are all busy would still have to wait when a specialist of that call type is idle. Therefore, this Bernoulli policy does not efficiently use the resources in the system, and leads to an inefficient one-step improved policy. To overcome the inefficiency of the Bernoulli policy, we instead use a policy that uses the specialists first, and assigns a call to a generalist only if a generalist is available while all specialists that can handle that call type are busy. The effectiveness of this policy has been rigorously justified in [9, 16].

Policy Evaluation

The relative value function for the policy $\hat{\pi}$, that uses specialists first and then generalists, is complicated. The policy $\hat{\pi}$ creates a dependence between the different agent groups that prohibit the derivation of a closed-form expression for the relative value function. Therefore, we approximate the relative value function by $\hat{V}$ as follows. Let $V_W(x,\lambda,\mu,S)$ be the relative value function of an M/M/S queueing system. The approximation $\hat{V}$ for the policy $\hat{\pi}$ is then given by

$$\hat{V}(\mathbf{q},\mathbf{x}) = \sum_{s\in\mathcal{S}} V_W(q_s+x_s,\tilde{\lambda}_s,\mu_{\{s\}},S_{\{s\}}) + V_W\big((q_1+x_1-S_1)^+ + \cdots + (q_N+x_N-S_N)^+ + x_{N+1},\lambda_{G_{N+1}},\mu_{G_{N+1}},S_{G_{N+1}}\big),$$

with $\tilde{\lambda}_s = \lambda_s\big(1-B(\lambda_s,\mu_{\{s\}},S_{\{s\}})\big)$ the approximate rate to the specialists of call type s, and $\lambda_{G_{N+1}} = \sum_{s\in\mathcal{S}} \lambda_s B(\lambda_s,\mu_{\{s\}},S_{\{s\}})$ the approximate rate to the generalists. Note that this approximation follows the idea of the motivating initial policy in that it ensures that all idle specialists of type s, given by $S_s - x_s$, are used for all calls in the queue of the same type. This results in $(q_s-[S_s-x_s])^+ = \max\{q_s+x_s-S_s,0\}$ calls that cannot be served. These calls are therefore waiting for a generalist to become idle. The approximation does take into account also the fact that a specialist can become idle before a generalist, and immediately assigns a call to the specialist.

The idea behind the approximation is that it roughly estimates the different flows to the different agent groups and then computes the value function as if the calls are waiting simultaneously at the two queues where they can be served. Note that strictly hierarchical architectures in which agents groups are structured so that no overflow of calls has to be split between two possible subsequent agent pools can be dealt with similarly. Observe that it might be possible that the overflow to the generalists is larger than their service rate. However, the system will still be stable

because the actual number of calls that will be served by the specialists will be higher, unless the system load is quite close to one.

Policy Improvement

In the policy improvement step, the initial policy $\hat{\pi}$ is improved by evaluating

$$\min\{\hat{V}(\mathbf{q}-e_s,\mathbf{x}+e_G)\,|\,s\in\mathcal{S},G\in\mathcal{G}_s,q_s>0,x_G<S_G\}\cup\{\hat{V}(\mathbf{q},\mathbf{x})\}.$$

Note that in this case the policy has to be determined for both the agent-assignment and the call-selection decisions.

The results for scenario 2 with also three skills are given in Table 2.6. The first line seems to suggest that no significant gains over the static policy can be obtained when the service rates of the specialists and the generalists are equal to each other. In Tables 2.7, 2.8, 2.9 we have varied the service rate of the generalists under different loads. The results show that as the system is offered higher loads, the gains by usingthe one-step improved policy are also higher as compared to the static routing

λ_X	μ_X	S_X	$\mu_{\{1,2,3\}}$	$S_{\{1,2,3\}}$	g	g'	g^*	Δ
6 6 6	2.0 2.0 2.0	3 3 3	2.0	3	11.043	10.899	10.746	1.429
6 3 3	1.5 1.0 1.0	3 3 3	1.0	3	20.122	19.259	18.361	4.892
6 5 4	2.5 2.0 1.5	2 2 2	2.0	3	13.186	11.562	11.314	2.187
6 5 4	1.8 1.8 1.2	3 3 3	1.2	3	16.348	14.931	14.602	2.253
7 6 5	2.5 2.0 1.5	3 3 3	2.0	2	15.269	13.310	13.127	1.397
7 6 5	1.8 1.8 1.2	3 3 3	1.8	3	23.260	20.091	19.125	5.054
10 6 3	3.0 2.0 1.0	3 3 3	1.0	2	31.834	26.844	25.184	6.594
6 5 4	3.0 2.0 1.0	3 3 3	1.0	2	20.083	14.403	13.795	4.404

Table 2.6: Numerical results for scenario 2

policy. Next, we scale the system such that the increase in the offered load was compensated by faster service rates or by an increase in the number of servers. Table 2.10 shows that when the service rates are scaled, the gains over the static policy remain roughly unaffected. However, when more servers are added, it slightly decreases. From the other lines in Table 2.6 we can conclude that the gain over the static policy is greater in asymmetric systems. In conclusion, we can observe that the improved policy has good performance and that its performance is close to the performance of the optimal policy.

Note that our proposed method is scalable, and can easily be used for bigger call centers. In this section, however, we restricted ourselves to a call center with three skills, since the computation of optimal policies becomes numerically difficult for bigger call centers. The optimal policy grows exponentially in the number of skills, while a single step of policy iteration has linear complexity.

$\mu_{\{1,2,3\}}$	g	g'	g^*	Δ
1.5	9.012	8.933	8.928	0.051
1.6	8.773	8.707	8.703	0.045
1.7	8.559	8.504	8.500	0.047
1.8	8.366	8.320	8.316	0.045
1.9	8.192	8.154	8.150	0.044
2.0	8.035	8.006	7.999	0.091
2.1	7.892	7.864	7.851	0.166
2.2	7.762	7.713	7.675	0.496
2.3	7.644	7.561	7.489	0.962
2.4	7.535	7.405	7.302	1.418
2.5	7.436	7.238	7.118	1.690

Table 2.7: Scenario 2—$\lambda_X = 5$, $\mu_X = 2$, $S_X = 3$, and $S_{\{1,2,3\}} = 3$

$\mu_{\{1,2,3\}}$	g	g'	g^*	Δ
1.5	13.674	13.142	12.877	2.055
1.6	12.995	12.595	12.344	2.035
1.7	12.405	12.106	11.872	1.969
1.8	11.891	11.665	11.453	1.848
1.9	11.440	11.265	11.080	1.671
2.0	11.043	10.899	10.746	1.429
2.1	10.692	10.449	10.445	0.046
2.2	10.379	10.179	10.160	0.189
2.3	10.101	9.934	9.881	0.532
2.4	9.851	9.712	9.610	1.062
2.5	9.626	9.494	9.347	1.568

Table 2.8: Scenario 2—$\lambda_X = 6$, $\mu_X = 2$, $S_X = 3$, and $S_{\{1,2,3\}} = 3$

$\mu_{\{1,2,3\}}$	g	g'	g^*	Δ
1.5	27.101	25.142	23.387	7.502
1.6	25.138	22.841	21.638	5.559
1.7	23.306	20.923	20.099	4.097
1.8	21.623	19.326	18.755	3.042
1.9	20.099	17.992	17.586	2.304
2.0	18.737	16.867	16.571	1.784
2.1	17.533	15.910	15.690	1.401
2.2	16.475	15.086	14.920	1.114
2.3	15.551	14.370	14.240	0.915
2.4	14.746	13.741	13.631	0.802
2.5	14.044	13.183	13.082	0.771

Table 2.9: Scenario 2—$\lambda_X = 7$, $\mu_X = 2$, $S_X = 3$, and $S_{\{1,2,3\}} = 3$

λ_X	μ_X	S_X	$\mu_{\{1,2,3\}}$	$S_{\{1,2,3\}}$	g	g'	g^*	Δ
6 5 4	2.5 2.0 1.5	2 2 2	2.0	3	13.186	11.562	11.314	2.187
12 10 8	5.0 4.0 3.0	2 2 2	4.0	3	13.187	11.564	11.313	2.215
24 20 16	10.0 8.0 6.0	2 2 2	8.0	3	13.190	11.567	11.310	2.273
12 10 8	2.5 2.0 1.5	4 4 4	2.0	6	18.625	17.789	17.562	1.290
24 20 16	5.0 4.0 3.0	4 4 4	4.0	6	18.628	17.792	17.559	1.326
24 20 16	2.5 2.0 1.5	8 8 8	2.0	12	31.925	31.380	30.991	1.253
30 20 10	2.5 2.0 1.5	15 15 15	2.0	15	28.924	27.996	27.256	2.714
30 20 20	2.5 2.0 1.5	20 20 20	2.0	20	35.290	34.829	32.015	8.788

Table 2.10: Scenario 2—scaling of the system

2.6 Application: A Controlled Polling System

The relative value function of the priority queue in Sect. 2.3.5 can be seen as a server assignment problem in which a fixed priority discipline is used, namely the preemptive priority resume policy. When the server can change at any instant, more freedom is introduced leading to lower average costs under the optimal policy. The question then becomes what the optimal policy is.

Let $c_1 > 0$ and $c_2 > 0$. In the case that the switching costs are identical to zero the optimal policy is equal to the well-known μc rule (see, e.g., [3]). This means that priority is given to a class-i customer based on the value of $\mu_i c_i$; priority is given to the queue with higher values of $\mu_i c_i$. When switching costs are greater than zero, the optimal policy is not known and a numerical method such as policy or value iteration has to be used. In this section we illustrate how to use one-step policy improvement for this problem.

In principle, one could apply a Bernoulli policy to the server, i.e., the server works a fraction α of the time on queue 1 and a fraction $1-\alpha$ on queue 2. However, in order to take the dependencies between the two queues into account in the initial policy, we can also use the priority policy, of which we denote the value function by V_p, of Sect. 2.3.5. Let $\lambda_1 = \lambda_2 = 1$, $\mu_1 = 6$, $\mu_2 = 3$, $c_1 = 2$, $c_2 = 1$, and $s_1 = s_2 = 2$. Note that with these parameters, the priority policy coincides with the μc rule. Define $\mu = \max\{\mu_1, \mu_2\}$. Then, define for some fixed x and y

$$z_{k,l} = s_k \mathbb{1}_{\{k \neq l\}} + c_1 x + c_2 y + \lambda_1 V_p(x+1,y,l) + \lambda_2 V_p(x,y+1,l) + \\ \mu_l V_p\big(((x,y)-e_l)^+, l\big) + (\mu - \mu_l) V_p(x,y,l)$$

as the expected bias if the server immediately switches from position k to position l and uses the preemptive priority rule afterwards. The one-step improved policy is simply the policy that minimizes for each (x,y,k) the expression $\min_a\{z_{k,a}\}$. Hence, the actions are defined by $a_{x,y,k} = \arg\min\{z_{k,a}\}$.

Table 2.11 shows the results for the one-step policy improvement. The average cost resulting from using a single step of policy iteration is 3.09895. This is a reduction of the costs by 14.6% as compared to using the μc rule where the average cost

is 3.62894. By using policy iteration to find the optimal policy the results hardly improve; the average cost is at lowest 3.09261. Surprisingly enough, the optimal policy is found in two steps of policy iteration. The fast convergence of the policy iteration algorithm is not coincidental; the average cost of the policies generated by policy iteration converge at least exponentially fast to the minimum cost, with the greatest improvement in the first few steps.

Iteration	Average cost	Comment
0	3.62894	μc rule
1	3.09895	One-step policy improvement
2	3.09261	Optimal policy

Table 2.11: Policy iteration results

References

1. E. Altman, Applications of Markov decision processes in communication networks: a survey, in *Handbook of Markov Decision Processes*, edited by E.A. Feinberg, A. Shwartz (Kluwer, Dordrecht, 2002)
2. S. Asmussen, O. Nerman, M. Olsson, Fitting phase type distributions via the EM algorithm. Scand. J. Stat. **23**, 419–441 (1996)
3. J.S. Baras, D.-J. Ma, A.M. Makowski, k competing queues with geometric service requirements and linear costs: the μc rule is always optimal. Syst. Control Lett. **6**, 173–180 (1985)
4. R. Bellman, Adaptive Control Processes: A Guided Tour (Princeton University Press, Princeton, NJ, 1961)
5. S. Bhulai, On the value function of the M/Cox(r)/1 queue. J. Appl. Probab. **43**(2), 363–376 (2006)
6. S. Bhulai, G.M. Koole, On the structure of value functions for threshold policies in queueing models. J. Appl. Probab. **40**, 613–622 (2003)
7. S. Bhulai, F.M. Spieksma, On the uniqueness of solutions to the Poisson equations for average cost Markov chains with unbounded cost functions. Math. Meth. Oper. Res. **58**(2), 221–236 (2003)
8. S. Bhulai, A.C. Brooms, F.M. Spieksma, On structural properties of the value function for an unbounded jump Markov process with an application to a processor-sharing retrial queue. Queueing Syst. **76**(4), 425–446 (2014)
9. P. Chevalier, N. Tabordon, R. Shumsky, Routing and staffing in large call centers with specialized and fully flexible servers. Working paper (2004)
10. E. Hyytiä, J. Virtamo, Dynamic routing and wavelength assignment using first policy iteration. Technical Report COST 257, Helsinki University of Technology (2000)
11. G.M. Koole, Stochastic Scheduling and Dynamic Programming. CWI Tract, vol. 113 (CWI, Amsterdam, 1995)

12. S.A. Lippman, Applying a new device in the optimization of exponential queueing systems. Oper. Res. **23**, 687–710 (1975)
13. R.A. Marie, Calculating equilibrium probabilities for $\lambda(m)/C_k/1/N$ queues, in *Proceedings of the International Symposium on Computer Performance Modeling* (1980)
14. R.E. Mickens, Difference Equations: Theory and Applications (Chapman & Hall, London, 1990)
15. J.M. Norman, Heuristic Procedures in Dynamic Programming (Manchester University Press, Manchester, 1972)
16. E.L. Örmeci, Dynamic admission control in a call center with one shared and two dedicated service facilities. IEEE Trans. Autom. Control **49**, 1157–1161 (2004)
17. T.J. Ott, K.R. Krishnan, Separable routing: a scheme for state-dependent routing of circuit switched telephone traffic. Ann. Oper. Res. **35**, 43–68 (1992)
18. M.L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming (Wiley, New York, 1994)
19. H. Rummukainen, J. Virtamo, Polynomial cost approximations in markov decision theory based call admission control. IEEE/ACM Trans. Netw. **9**(6), 769–779 (2001)
20. S.A.E. Sassen, H.C. Tijms, R.D. Nobel, A heuristic rule for routing customers to parallel servers. Statistica Neerlandica **51**, 107–121 (1997)
21. R.F. Serfozo, An equivalence between continuous and discrete time Markov decision processes. Oper. Res. **27**, 616–620 (1979)
22. F.M. Spieksma, Geometrically ergodic Markov chains and the optimal control of queues. Ph.D. thesis, Leiden University (1990)
23. S. Stidham Jr., R.R. Weber, A survey of Markov decision models for control of networks of queues. Queueing Syst. **13**, 291–314 (1993)
24. H.C. Tijms, Stochastic Models: An Algorithmic Approach (Wiley, New York, 1994)
25. D. Towsley, R.H. Hwang, J.F. Kurose, MDP routing for multi-rate loss networks. Comput. Netw. ISDN **34**, 241–261 (2000)

Chapter 3
Approximate Dynamic Programming by Practical Examples

Martijn R.K. Mes and Arturo Pérez Rivera

Abstract Computing the exact solution of an MDP model is generally difficult and possibly intractable for realistically sized problem instances. A powerful technique to solve the large scale discrete time multistage stochastic control processes is Approximate Dynamic Programming (ADP). Although ADP is used as an umbrella term for a broad spectrum of methods to approximate the optimal solution of MDPs, the common denominator is typically to combine optimization with simulation, use approximations of the optimal values of the Bellman's equations, and use approximate policies. This chapter aims to present and illustrate the basics of these steps by a number of practical and instructive examples. We use three examples (1) to explain the basics of ADP, relying on value iteration with an approximation of the value functions, (2) to provide insight into implementation issues, and (3) to provide test cases for the reader to validate its own ADP implementations.

Key words: Dynamic programming, Approximate dynamic programming, Stochastic optimization, Monte Carlo simulation, Curse of dimensionality

3.1 Introduction

Approximate Dynamic Programming (ADP) is a powerful technique to solve large scale discrete time multistage stochastic control processes, i.e., complex Markov Decision Processes (MDPs).

M.R.K. Mes (✉) • A. Pérez Rivera
Department of Industrial Engineering and Business Information Systems, University of Twente, Enschede, The Netherlands
e-mail: m.r.k.mes@utwente.nl

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_3

MDPs consist of a state space $\mathcal{S}$, and at each time step t, the system is in a particular state $S_t \in \mathcal{S}$ from which we can take a decision x_t from the feasible set $\mathcal{X}_t$. This decision results in rewards or costs, typically given by $C_t(S_t, x_t)$, and brings us to a new state S_{t+1} with probability $\mathbb{P}(S_{t+1}|S_t, x_t)$, i.e., the next state is conditionally independent of all previous states and actions. Therefore, the decision not only determines the direct costs, but also the environment within which future decisions take place, and hence influences the future costs. The goal is to find a policy. A policy $\pi \in \Pi$ can be seen as a decision function $X^\pi(S_t)$ that returns a decision $x_t \in \mathcal{X}_t$ for all states $S_t \in \mathcal{S}$, with Π being the set of potential decision functions or policies. The problem of finding the best policy can be written as

$$\min_{\pi \in \Pi} \mathbb{E}^\pi \left\{ \sum_{t=0}^{T} \gamma C_t(S_t, X_t^\pi(S_t)) \right\}, \tag{3.1}$$

where γ is a discount factor (minimizing the total and average rewards is achieved by setting $\gamma = 1$ and $\gamma = 1/T$ respectively), and T denotes the planning horizon (could be infinite).

The problem formulation (3.1) covers a huge variety of decision problems, found in various applications also covered in this book, such as health care, production and manufacturing, communications, and transportation. There is also a wide variety of methods to solve (3.1), such as rolling-horizon procedures, simulation optimization, linear programming, and dynamic programming. Here, we focus on the latter.

Dynamic programming solves complex MDPs by breaking them into smaller subproblems. The optimal policy for the MDP is one that provides the optimal solution to all sub-problems of the MDP [1]. This becomes visible in Bellman's equation, which states that the optimal policy can be found by solving:

$$V_t(S_t) = \min_{x_t \in \mathcal{X}_t} \left(C_t(S_t, x_t) + \gamma \sum_{s' \in \mathcal{S}} \mathbb{P}(S_{t+1} = s'|S_t, x_t) V_{t+1}(s') \right). \tag{3.2}$$

Computing the exact solution, e.g., by using backward dynamic programming, is generally difficult and possibly intractable for large problems due to the widely known "curse of dimensionality". One can encounter three curses of dimensionality in dynamic programming [10]:

1. the state space $\mathcal{S}$ for the problem may be too large to evaluate the value function $V_t(S_t)$ for all states within reasonable time;
2. the decision space $\mathcal{X}_t$ may be too large to find the optimal decision for all states within reasonable time;
3. computing the expectation of 'future' costs may be intractable when the outcome space is large.

The outcome space is the set of possible states in time period $t+1$, given the state and decision in time period t. Its size is driven by the random information W_{t+1} that arrives between t and $t+1$. By capturing all the randomness with W_{t+1}, we can express the next state S_{t+1} as a function of the current state S_t, the decision x_t, and the new information W_{t+1}, using a transition function $S_{t+1} = S^M(S_t, x_t, W_{t+1})$. Now, assume that W_{t+1} is independent on all prior information, and let Ω_{t+1} be the set of

possible outcomes of W_{t+1} and let $\mathbb{P}(W_{t+1} = \omega)$ denote the probability of outcome $\omega \in \Omega_{t+1}$. We now rewrite (3.2) as

$$V_t(S_t) = \min_{x_t \in \mathcal{X}_t} \left(C_t(S_t, x_t) + \gamma \sum_{\omega \in \Omega_{t+1}} \mathbb{P}(W_{t+1} = \omega) V_{t+1}(S_{t+1} | S_t, x_t, \omega) \right), \quad (3.3)$$

with $S_{t+1} = S^M(S_t, x_t, \omega)$. Note that in (3.2) and (3.3) the direct costs $C_t(S_t, x_t)$ are assumed to be deterministic; however, the random information ω could also play a role in this function, see [10].

Based on an MDP model, Approximate Dynamic Programming (ADP) is a modeling framework that offers several strategies for tackling the curses of dimensionality in large, multi-period, stochastic optimization problems [10]. Also for ADP, the output is a policy or decision function $X_t^\pi(S_t)$ that maps each possible state S_t to a decision x_t, for each stage t in the planning horizon. Although ADP is used as an umbrella term for a broad spectrum of methods to approximate the optimal solution of MDPs, the common denominator is typically to combine optimization with simulation (sampling from Ω_{t+1}), use approximations of the optimal values of the Bellman's equations, and use approximate policies. For applications of ADP, a more natural form of the Bellman's equations in (3.3) is the expectational form given by:

$$V_t(S_t) = \min_{x_t \in \mathcal{X}_t} \left(C_t(S_t, x_t) + \gamma \mathbb{E}^\omega \left\{ V_{t+1}(S_{t+1} | S_t, x_t, \omega) \right\} \right). \quad (3.4)$$

To approximate the optimal values of (3.4), a series of constructs and algorithmic manipulations of the MDP model are needed. This chapter aims to present and illustrate the basics of these steps by a number of practical and instructive examples.

ADP can be applied to large-scale instances because it is able to handle two of the three dimensionality issues. First, the large outcome space can be handled through the construct of a post-decision state $S_t^{x,n}$, which we explain in Sect. 3.2. Second, the large state space can be handled by (1) generating sample paths through the states, the so-called "forward dynamic programming" algorithmic strategy, which solves the Bellman's equations by stepping forward in time, and repeating this process for N iterations, and (2) using approximate value functions $\overline{V}_t^n(S_t^{x,n})$ that are "learned" through the iterations and that might allow generalization across states, i.e., instead of learning the values of each state individually, visited states might tell us something about the value of states not visited yet. We also elaborate on this strategy in the next sections.

There are numerous ADP methods and variations available, each having their merits. Variations particularly consists in the type of value function approximations (e.g., lookup, parameterized, statistical) and policies used (e.g., policy approximations, tree search, roll-out heuristics, rolling horizon policies). In Sect. 3.5 we discuss some of these variations. But first, we explain the basic concept of ADP by means of three example problems. We use the examples (1) to explain the basics of ADP, relying on value iteration with an approximation of the value functions, (2) to provide insight into implementation issues, and (3) to provide test cases for the reader to validate its own ADP implementations (for the first two examples, we provide all experimental settings, ADP results, as well as the exact results). For each

of the three examples we use a similar division, consisting of problem introduction, the MDP model with exact results, and the ADP approach. We introduce the basics of ADP using a first transportation example in Sect. 3.2. Next, we present ADP extensions using another transportation example in Sect. 3.3 and using an example application of ADP in healthcare in Sect. 3.4. As an aid to the reader, the notation used throughout these examples is summarized in Table 3.1.

Variable	Description
$S \in \mathcal{S}$	State
$x \in \mathcal{X}$	Decision
$\pi \in \Pi$	Policy
$X^{\pi}(S_t)$	Decision function returning a decision in state S_t under π
$C_t(S_t, x_t)$	Costs function giving the costs of decision x_t in state S_t
γ	Discount factor
T	Length planning horizon
$\mathbb{P}(S_{t+1}\|S_t, x_t)$	Transition probability to S_{t+1} given S_t and x_t
$W_{t+1} \in \Omega_{t+1}$	Random information that arrives between t and $t+1$
$V_t(S_t)$	Value function
$S^m(S_t, x_t, W_{t+1})$	Transition function to S_{t+1} given S_t, x_t, and W_{t+1}
S_t^x	Post-decision state
$S^{M,x}(S_t, x_t)$	Transition function to S_t^x given S_t and x_t
$\overline{V}_t^n(S_t^{x,n})$	Approximation of the value of post-decision state $S_t^{x,n}$
$\tilde{x}_t^n$	Expected optimal decision within the current state
$\hat{v}_t^n$	Approximation of the value of the current state given $\tilde{x}_t^n$
$U^V(\overline{V}_{t-1}^{n-1}(S_{t-1}^{x,n}), S_{t-1}^{x,n}, \hat{v}_t^n)$	Function to update the approximation $\overline{V}_{t-1}^n(S_{t-1}^{x,n})$
α	Stepsize used in the updating function U^V
$n = 1, \ldots, N$	Iteration counter
M	Number of iterations between "side simulations"
O	Number of iterations of a "side simulation"
K	Number of replications over all iterations N
$a \in \mathcal{A}$	Features
θ_a^n	Weight of feature a
$\phi_a\left(S_t^{x,n}\right)$	Basis function corresponding to feature a

Table 3.1: Frequently used notation throughout this chapter

3.2 The Nomadic Trucker Example

First, we briefly introduce the problem in Sect. 3.2.1 after which we present the MDP model (Sect. 3.2.2) and the ADP approach (Sect. 3.2.3).

3.2.1 Problem Introduction

The nomadic trucker problem is a stylized transportation problem in which a single trucker observes demands that arise randomly in different locations and he moves between these locations to accept those loads that maximize the long-term reward. This problem, which is similar to the well known taxicab problem, is described in [3, 4], and [10]. Here, we slightly modify the problem settings to allow repeatability of the experiments without having to provide extensive data sets.

Our trucker is characterized by its current location $l_t \in \mathcal{L}$ (set of 256 locations), the day of the week $d_t \in \{1,\ldots,7\}$ (Monday till Sunday), and its trailer type $k_t \in \{1,\ldots,3\}$ (small, medium, and large trailer). Every day, the driver observes loads to move from its current location to another location. The daily decision from a given location i is which location j to visit next, either through a loaded move (when a load from i to j is available) or an empty move, or to stay at the current location. After the decision, loads that are not selected are lost (assumed to be picked up by others). Further, it is assumed that all moves take less then a day, i.e., the next day a new decision has to be made.

The probability that, on a given day of the week d, a load from i to j will appear is given by p_{ij}^d (see the Appendix). The trailer type attribute varies in a cyclic fashion, irrespective of the remaining attributes. For example, if at time period t the trailer type attribute is large, then at time $t+1$ the trailer type will be small, and at time $t+2$ it will be medium. A larger trailer type results in higher rewards when traveling loaded or costs when traveling empty. We use the rewards/costs $c(k) = (1, 1.5, 2)$ per unit distance, for $k = 1,\ldots,3$. The rewards for loads leaving location i are further multiplied by the origin probability b_i. The distance between i and j is given by $d(i,j)$.

We denote the described instance where there driver is characterized by a location, trailer type, and day of the week as the multi-attribute version. For the single-attribute version, we omit the trailer type and day of the week and use the settings for trailer type 1 and day of the week 1, i.e., $c(k) = 1$ and $p^d = 1$.

3.2.2 MDP Model

We subsequently present the following elements of the MDP model: the state (Sect. 3.2.2.1), the decision (Sect. 3.2.2.2), the costs (Sect. 3.2.2.3), the new information and transition function (Sect. 3.2.2.4), and the solution (Sect. 3.2.2.5).

3.2.2.1 State

The state S_t consists of resource and demand information: $S_t = \{R_t, D_t\}$. R_t is a vector of attributes (l_t, d_t, k_t) representing the current location of the trucker, the current day of the week, and the trailer type, respectively. D_t is a vector indicating

for each location $i \in \mathcal{L}$ whether there is a load available from l_t to i ($D_{t,i} = 1$) or not ($D_{t,i} = 0$). The state contains all the information necessary to make decisions; in this case the resource and demand information. The size of the state space is given by the number of possible settings of the resource vector R_t, which is 5376 ($256 \times 7 \times 3$), times the number of possible load realizations, which is 2^{256}.

3.2.2.2 Decision

The decision x_t provides the location j where we want to go to. Note that, given the used cost structure, if we decide to go from l_t to j and there is a load available from l_t to j, it does not make sense to travel empty. In other words, from the demand vector D_t we can infer whether the decision to go to location j will imply an empty or a loaded move. Hence, it is sufficient to describe the decision x_t with the location j, meaning the decision space $\mathcal{X}_t$ equals $\mathcal{L}$, with size 256.

3.2.2.3 Costs

The costs of decision x_t are given by

$$C(S_t, x_t) = \begin{cases} -c(k_t)d(l_t, x_t), & \text{if } D_{t,x_t} = 0. \\ c(k_t)d(l_t, x_t)b_i, & \text{if } D_{t,x_t} = 1. \end{cases} \tag{3.5}$$

3.2.2.4 New Information and Transition Function

After making decision x_t and before arrival in state S_{t+1}, new information W_{t+1} arrives. Here, the new information gives the load availability at time $t+1$. The transition from S_t to S_{t+1} is given by

$$S_{t+1} = S^M(S_t, x_t, W_{t+1}), \tag{3.6}$$

where $l_{t+1} = x_t$, $d_{t+1} = d_t \pmod 7 + 1$, $k_{t+1} = k_t \pmod 3 + 1$, and $D_{t+1} = W_{t+1}$. The size of the outcome space is given by all possible load realizations: 2^{256}.

3.2.2.5 Solution

The objective in this example is to maximize profit. Therefore, we need to replace the minimization objective in (3.3) by a maximization objective. However, to be consistent in our presentation, we use the minimization term throughout this chapter.

Even for this toy problem, the state space and the outcome space is already large. To ease the computation, we solve the problem for all possible "resource states", i.e.,

for each resource state R_t at each stage t, we calculate its expected value considering all possible load availabilities with their probabilities. This can be seen as a "post-decision" state as we introduce in Sect. 3.2.3.1. Once the values for all "resource states" are determined, we can easily derive the optimal decision from each "full" state using (3.4) together with the transition function (3.6), where the transition only considers the change from the "full" state to the "resource state", i.e., the new information W_{t+1} does not play a role in this transition.

For the exact computation of the values of the "resource states", it is not necessary to evaluate all possible permutations of the load combinations W_{t+1}. Within a given resource state R_t, we can rank on forehand the $2 \times |\mathcal{L}| = 512$ possible decisions (loaded and empty moves). We then start from the best possible decision and multiply its corresponding probability (if the decision involves a loaded move, we use the probability of having the corresponding load available, otherwise we use a probability of one) with its value; with one minus the before mentioned probability, we consider the second best possible decision and so on. We sum up all probabilities times the values to compute the expected value under the optimal policy.

We compute the optimal solution for three cases: (1) the infinite horizon single-attribute case, (2) the infinite horizon multi-attribute case, and (3) the finite horizon single-attribute case. For the finite horizon case, we can easily compute the value functions using backwards dynamic programming with (3.3). For the infinite horizon cases, we use value iteration to determine the optimal values. For the multi-attribute case, we use as initial state $S_0 = (1,1,1)$ and for the single-attribute cases we use $S_0 = (1)$. Further, for the finite horizon case, we use a discount $\gamma = 1$ and a horizon length $T = 20$, and for the infinite cases we use $\gamma = 0.9$. The optimal values are: (1) 8364.31 for the infinite horizon single-attribute case, (2) 11,448.48 for the infinite horizon multi-attribute case, and (3) 17,491.95 for the finite horizon single-attribute case.

3.2.3 Approximate Dynamic Programming

Even though the introduced version of the nomadic trucker problem is a simplified problem that can easily be solved exactly, it allows us to introduce the basics of ADP. We introduce the concept of a post-decision state (Sect. 3.2.3.1), the forward dynamic programming approach (Sect. 3.2.3.2), and the use of value function approximations (Sect. 3.2.3.3). We give a typical outline of an ADP algorithm and present experimental results throughout Sects. 3.2.3.2 and 3.2.3.3.

3.2.3.1 Post-decision State

The post-decision state, represented by S_t^x, is the state immediately after action x_t, but before the arrival of new information W_{t+1}. The information embedded in the post-decision state allows us to estimate the downstream costs. We can assign the

expected downstream costs $\mathbb{E}^{\omega}\{V_{t+1}(S_{t+1}|S_t^x,\omega)\}$ to every post-decision state S_t^x, thereby eliminating the need to evaluate all possible outcomes ω for every action. Consider the following optimality equations:

$$V_{t-1}^x(S_{t-1}^x) = \mathbb{E}^{\omega}\{V_t(S_t|S_{t-1}^x,\omega)\} \tag{3.7}$$

$$V_t(S_t) = \min_{x_t\in\mathcal{X}_t}\left(C_t(S_t,x_t)+\gamma V_t^x(S_t^x)\right) \tag{3.8}$$

$$V_t^x(S_t^x) = \mathbb{E}^{\omega}\{V_{t+1}(S_{t+1}|S_t^x,\omega)\} \tag{3.9}$$

If we substitute (3.9) into (3.8), we obtain the standard form of Bellman's equation (3.4). However, if we substitute (3.8) into (3.7), we obtain the optimality equations around the post-decision state variable

$$V_{t-1}^x\left(S_{t-1}^x\right) = \mathbb{E}^{\omega}\left\{\min_{x_t\in\mathcal{X}_t}\left(C_t\left(S_t,x_t\right)+\gamma V_t^x(S_t^x|S_{t-1}^x,\omega)\right)\right\}. \tag{3.10}$$

The basic idea now is (1) to use the deterministic optimization problem of (3.8) to make decisions and (2) to use the resulting observations to update an estimate $\overline{V}_{t-1}^{n-1}(S_{t-1}^x)$ of $V_{t-1}^x(S_{t-1}^x)$ thereby approximating the expectation in (3.10). We update the estimates $\overline{V}_{t-1}^{n-1}(S_{t-1}^x)$ iteratively over a number of iterations n, each consisting of a Monte Carlo simulation of the random information ω, which will be further explained in Sect. 3.2.3.2.

We express our transition from state S_t with action x_t to the post-decision state S_t^x by

$$S_t^x = S^{M,x}(S_t,x_t). \tag{3.11}$$

For our example, the post-decision state S_t^x is determined as if we had already arrived at the destination x_t at time t. That is, we change the location, day of the week, and time components of the state to represent the day when we will be at the chosen location: $l_t^x = x_t$, $d_t^x = d_t \pmod 7 + 1$, and $k_t^x = k_t \pmod 3 + 1$. Note that, although the concept of a post-decision state is generally used in the context of ADP only, we already used it to calculate the exact solution of the MDP (see Sect. 3.2.2.5). An illustration of the transition of states can be found in Fig. 3.1.

3.2.3.2 Forward Dynamic Programming

In the forward dynamic programming algorithmic strategy, the Bellman's equations are solved only for one state at each stage, using estimates of the downstream values, and performing iterations n to learn these downstream values. To make clear we are dealing with iterations, we add a superscript n to the decisions and state variables.

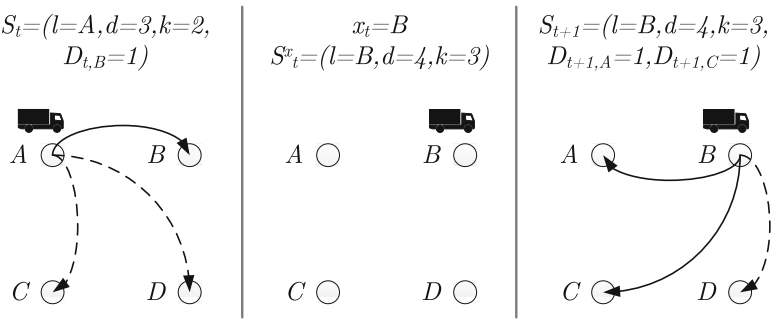


Fig. 3.1: Transition of states in the nomadic trucker problem

We introduce the construct of approximated next-stage costs (estimated downstream values) $\overline{V}_t^n(S_t^{x,n})$, which replaces the standard expectation in Bellman's equations, see (3.9), with an approximation

$$\overline{V}_t^n(S_t^{x,n}) = \mathbb{E}^{\omega}\left\{V_{t+1}\left(S_{t+1}^n|S_t^{x,n},\omega\right)\right\}. \tag{3.12}$$

Using the post-decision state and the approximated next-stage cost, the original Bellman's equations from (3.4) are converted to the ADP forward optimality equations, as seen in (3.13).

$$\hat{v}_t^n = \min_{x_t^n \in \mathcal{X}_t}\left(C\left(S_t^n,x_t^n\right) + \gamma \overline{V}_t^{n-1}\left(S^{M,x}\left(S_t^n,x_t^n\right)\right)\right). \tag{3.13}$$

The decision that minimizes (3.13) is given by

$$\tilde{x}_t^n = \arg\min_{x_t^n \in \mathcal{X}_t}\left(C\left(S_t^n,x_t^n\right) + \gamma \overline{V}_t^{n-1}\left(S^{M,x}\left(S_t^n,x_t^n\right)\right)\right). \tag{3.14}$$

Note that $\tilde{x}_t^n$ is a pure exploitation decision, i.e., the decision for which we currently expect it gives the best results. Given that the decision $\tilde{x}_t^n$ relies on the approximation $\overline{V}_t^{n-1}\left(S^{M,x}\left(S_t^n,x_t^n\right)\right)$, the decision might not be optimal with respect to the MDP solution. Further, as we will show later on, policies other than pure exploitation might be useful.

For each feasible decision x_t^n, there is an associated post-decision state $S_t^{x,n}$ obtained using (3.11). The ADP forward optimality equations are solved first at stage $t = 0$ for an initial state S_0, and then for subsequent stages and states until the end of the horizon for the finite horizon case, or a predetermined number of iterations for the infinite horizon case. In each iteration n, a sample path $\omega^n \in \Omega$ is drawn, with Ω being the set of all sample paths. We use $W_t(\omega^n)$ to denote the sample realization at time t using the sample path ω^n in iteration n. To advance "forward" in time, from stage t to $t+1$, the sample $W_{t+1}(\omega^n)$ is used. With this information, transition in the algorithm is done using the same transition as in the MDP model, see (3.6).

Immediately after the forward optimality equations are solved, the approximated next-stage cost $\overline{V}_t^{n-1}(S_t^{x,n})$ is updated retrospectively. The rationale behind this update is that, at stage t, the algorithm has seen new arrival information (via the simulation of ω^n) and has taken a decision in the new state S_t^n that incurs a cost. This means that the approximated next-stage cost that was calculated at the previous stage $t-1$, i.e., $\overline{V}_{t-1}^{n-1}(S_{t-1}^{x,n})$, has now been observed at stage t. To take advantage of this observation and improve the approximation, the algorithm updates this approximated next-stage cost of the previous post-decision state $S_{t-1}^{x,n}$ using the old approximation, i.e., $\overline{V}_{t-1}^{n-1}(S_{t-1}^{x,n})$ and the new approximation, i.e., the value $\hat{v}_t^n$ given by (3.13). We introduce U^V to denote the process that takes all of the aforementioned parameters and "tunes" the approximating function as follows:

$$\overline{V}_{t-1}^{n}(S_{t-1}^{x,n}) \leftarrow U^V(\overline{V}_{t-1}^{n-1}(S_{t-1}^{x,n}), S_{t-1}^{x,n}, \hat{v}_t^n) \tag{3.15}$$

Using all ingredients mentioned in this section, the ADP algorithm consists of looping over iterations $n = 1,\ldots,N$, and within each iteration, sequentially solving a subproblem for each $t \in \mathcal{T}$, using sample realizations of W_t, and updating our approximation of 'future' costs with (3.15). Consecutively, the subproblems are solved using the updated value function approximations in the next iteration. The output of the algorithm is the approximation $\overline{V}_t^N(S_t^x)$, for all $t \in \mathcal{T}$, which can be used to find the best decision for each time period and each state.

A typical outline of an ADP algorithm is shown in Algorithm 1. The infinite horizon version of this algorithm is basically the same, except (1) all subscripts t are removed, (2) ω^n represents a single sample instead of a sample path, (3) the loop over t is removed, (4) the condition "if $t > 0$" is removed, and (5) the previous post-decision state is the one from the previous iteration, $S^{x,n-1}$.

Algorithm 1 *Approximate Dynamic Programming algorithm*

Step 0. Initialization

- Step 0a. Choose an initial approximation $\overline{V}_t^0 \forall t \in \mathcal{T}$.
- Step 0b. Set the iteration counter $n = 1$, and set the maximum number of iterations N.
- Step 0c. Set the initial state to S_0^1.

Step 1. Choose a sample path ω^n.

Step 2. Do for $t = 0,\ldots,T$:

- Step 2a. Solve (3.13) to get $\hat{v}_t^n$ and (3.14) to get $\tilde{x}_t^n$.
- Step 2b. If $t > 0$, then update the approximation $\overline{V}_{t-1}^n\left(S_{t-1}^{x,n}\right)$ for the previous post-decision $S_{t-1}^{x,n}$ state using (3.15).
- Step 2c. Find the post-decision state $S_t^{x,n}$ with (3.11) and the new pre-decision state S_{t+1}^n with (3.6).

Step 3. Increment n. If $n \leq N$ go to Step 1.

Step 4. Return the value functions $\overline{V}_t^N\left(S_t^{x,n}\right) \forall t \in \mathcal{T}, S_t \in \mathcal{S}$.

Algorithm 1 relies on classical approximate value iteration with a pure forward pass. This means that at each step forward in time in the algorithm, the value function approximations are updated. As the algorithm steps forward in time, it may take many iterations before the costs incurred in later time periods are correctly transferred to the earlier time periods. To overcome this, the ADP algorithm can also be used with a double pass approach [10], consisting of a forward pass and a backward pass. In the forward pass, we simulate decisions moving forward in time, remembering the trajectory of states, decisions, and outcomes. Then, in a backward pass, we update the value functions moving backwards in time using the trajectory information. For the double pass algorithm, we remove the computation of $\hat{v}_t^n$ from Step 2a, delete Step 2b, and add an extra step just before Step 3 (renaming original Step 3 and Step 4 by Step 4 and Step 5 respectively) given in Algorithm 2:

Algorithm 2 *Backward pass to be used in the ADP algorithm*

Step 3. Do for $t = T, T-1, \ldots, 1$:

Step 3a. Compute $\hat{v}_t^n$ using the decision $\tilde{x}_t^n$ from the forward pass:

$$\hat{v}_t^n = C_t(S_t^n, \tilde{x}_t^n) + \gamma \hat{v}_{t+1}^n, \text{ with } \hat{v}_{T+1}^n = 0.$$

Step 3b. Update the approximation $\overline{V}_{t-1}^n\left(S_{t-1}^{x,n}\right)$ for the previous post-decision state $S_{t-1}^{x,n}$ using (3.15).

Computational Results

We now illustrate the working of this basic algorithm using the three variants of the nomadic trucker problem: infinite horizon single-attribute, infinite horizon multi-attribute, and finite horizon single-attribute. For the updating function we use a fixed stepsize $\alpha = 0.05$ given by $U^V(\overline{V}_{t-1}^{n-1}(S_{t-1}^{x,n}), S_{t-1}^{x,n}, \hat{v}_t^n) = (1-\alpha)\overline{V}_{t-1}^{n-1}(S_{t-1}^{x,n}) + \alpha \hat{v}_t^n$.

We show two performance measures of the ADP algorithm. First, we show the estimate $\overline{V}_0^n\left(S_0^{x,n}\right)$ of the initial state $S_0^{x,n}$ for different number of iterations n. Next, we show the discounted rewards of using the estimates for different number of iterations n. More precisely, for a given number of iterations n, we perform a simulation on the side. Each of these simulations uses O iterations, fixing the value function estimates and following the policy that uses these values (basically following Algorithm 1 with the initial approximation $\overline{V}_0^n$ resulting from the past n iterations and skipping Step 2b). We perform the simulation every Mth iteration, i.e., for $n = M, 2M, \ldots, N$. Finally, to provide representative results, we repeat the whole procedure K times and report the averages. The used settings for N, M, O, and K are shown in the figure captions.

The results of the basic ADP algorithm is shown in Fig. 3.2 under the policy "Expl-F", since we follow the pure exploitation policy (3.13) with a fixed stepsize (F). In addition, we show the results using two other stepsizes. First, the harmonic stepsize (H) given by $\alpha^n = \max\left\{\frac{\lambda}{\lambda+n-1}, \alpha^0\right\}$, with $\lambda = 25$ and $\alpha^0 = 0.05$. Second, the BAKF stepsize (B), the bias adjusted Kalman Filter also known as OSA (Optimal Stepsize Algorithm). For a review on stepsizes we refer to [3] and [10, Chap. 11]. The policy "OPT" refers to the optimal value.

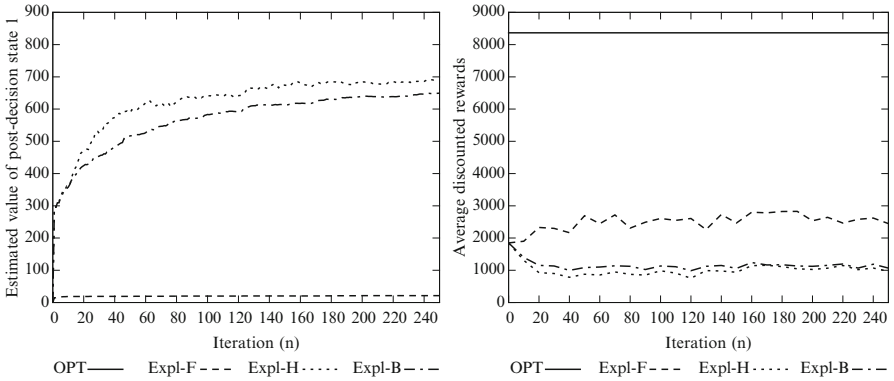


Fig. 3.2: Infinite horizon single-attribute case: resulting estimate $\overline{V}_0^n\left(S_0^{x,n}\right)$ (*left*) and realized rewards (*right*), using $N = 250$, $M = 10$, $O = 1000$, and $K = 100$

Clearly, the value function estimate for the initial state is still far off: more then 90% away from the optimal value. Especially, the fixed stepsize gives terrible results. The explanation is that because we initialized all values at zero, the first observation only increases the estimate with 5% of the observed value. The other two stepsizes start with $\alpha^1 = 1$, resulting in a faster increase in estimated value. However, when we look at the performance, the "better" value function estimates result in worse performance. The explanation is that after a couple of iterations, some states have been visited more often than other states. As a result, for some states we might still use the initialized approximations (in our case values of zero) whereas for other states we have a reasonable estimate. When making the pure exploitation decision, we now tend to visit the states we have visited before, even if this would result in relatively high direct costs. In this case, we perform worse compared to a myopic policy. The results for following the myopic policy can be found at $n = 0$ in the right figure, since we then perform a simulation with the initial approximation of having zero downstream costs. Although the results are caused by some simplifications within the used ADP algorithm, the resulting phenomenon might also be present in more advanced ADP implementations, since the decision what state to measure next might still be biased by the number of measurements of the different states.

In general, using the ADP algorithm as shown before would not work. Most of the time, we need to make some modifications. First, within our ADP algorithm we need to make a tradeoff between exploitation, i.e., making the best decision based on the current value function estimates using (3.13), and exploration to learn the value of states frequented less often. Second, we need a value function approximation that is able to generalize across states, i.e., an observation not only results in an update for the value of the corresponding state, but also of other states. In the remainder of this section, we briefly touch upon the exploration issue. The issue of having a better value function approximation is discussed in Sect. 3.2.3.3.

To overcome the problem of limited exploration, we might enforce exploration by stating that a certain fraction of the time, say ε, the policy should perform exploration using a random decision instead of exploitation using the decision $\tilde{x}_t^n$ from (3.14). This decision policy is known as ε-greedy. When making an exploration decision, it is likely that $x_t^n \neq \tilde{x}_t^n$. We now have a choice whether we use $\hat{v}_t^n$ (corresponding to decision $\tilde{x}_t^n$) to update $\overline{V}_{t-1}^n\left(S_{t-1}^{x,n}\right)$ or to use the estimated costs corresponding with the actual decision x_t^n, i.e., $C\left(S_t^n, x_t^n\right) + \gamma \overline{V}_t^{n-1}\left(S^{M,x}\left(S_t^n, x_t^n\right)\right)$. The policy to determine the decisions on what state to visit next is often referred to as the behavior or sampling policy. The policy that determines the decision that seems to be the best, i.e., using (3.14) is denoted by learning policy. When the sampling policy and the learning policy are the same, this is called on-policy learning, otherwise it is called off-policy learning. In most cases, off-policy learning results in faster convergence; but there are cases where on-policy learning is preferred, see [10, Chap. 9] for more information. In the remainder, unless stated differently, we use off-policy learning.

The results of the ε-greedy policies are shown in Fig. 3.3 under the policy "Epsε-S", where $\varepsilon = 0.25$ or $\varepsilon = 1$, and S denotes the stepsize (H for harmonic and B for BAKF). The policies Hepsε will be introduced later on.

From Fig. 3.3, we see that the ε-greedy policies improve the performance, both with respect to convergence of value functions (estimated values of the initial post-decision state) and the average discounted rewards, when compared to the exploitation policies. However, we also see that policies with faster convergence in value function not necessarily yield better performance. Again, this phenomenon will also be present at more advanced ADP implementations. First, having a good approximation for one state does not necessarily mean we have a good approximation for other states. Particularly when using value function approximations that are able to generalize across states, we might have states that we consistently underestimate and others that we consistently overestimate. Second, the (relative) ordering of states might already result in good performance of the policy (i.e., decision function) itself. In this example, the absolute values of the downstream costs might be less important when choosing between decisions with similar direct costs.

Still, we observe that our estimate is far off from the optimal value and we achieve only about 68% of the rewards under the optimal policy. To further improve the learning rate, i.e., increase the performance using the same number of iterations,

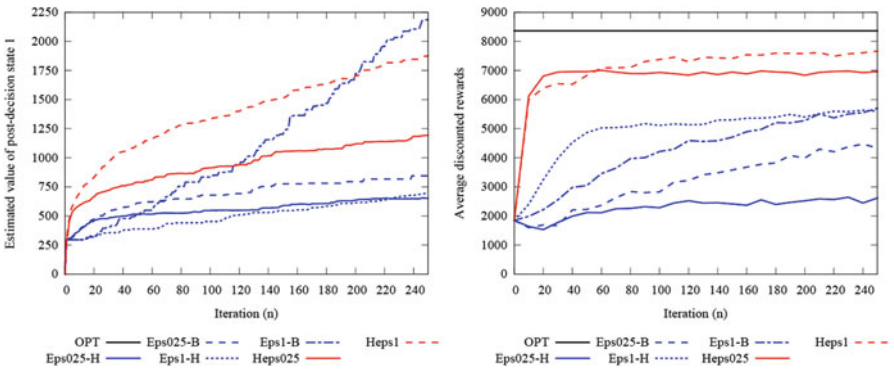


Fig. 3.3: Infinite horizon single-attribute case: resulting estimate $\overline{V}_0^n\left(S_0^{x,n}\right)$ (*left*) and realized rewards (*right*), using $N = 250$, $M = 10$, $O = 1000$, and $K = 100$

we are going to use a Value Function Approximation (VFA) that is able to generalize across states. There are many options for this, in this chapter we present two specific forms: hierarchical state aggregation and a basis function approach. For the nomadic trucker problem, we illustrate the state aggregation approach.

3.2.3.3 Value Function Approximation

Although adopting the concept of the post-decision state greatly reduced the computational burden, (3.10) still requires a post-decision state to be visited sufficiently often in order to learn about its associated downstream costs, which would not be possible for realistic sized problem instances. The reason for this is that the value function approximation used in the previous section is updated for one state per stage per iteration. This approach is known as the lookup-table approach. A good value function approximation is able to generalize across states, such that an observation for one state results in an update of the value of many states.

For this problem, we use an hierarchical aggregation approach as presented in [4]. A standard strategy in ADP is to aggregate the state space. Each state belongs to an aggregated state, and instead of estimating the value of states, we estimate the value of aggregate states consisting of multiple states. However, aggregation requires resolving the classic tradeoff between aggregation error and sampling error. In the hierarchical aggregation approach, we use multiple aggregation levels and each observation is used to update aggregate states at different aggregation level. The value of a state is estimated using a weighted combination of the value of the aggregated states this state belongs to.

For this example, we use the following hierarchical aggregation structure. With respect to the state variables trailer type k and day of the week d, we either include them or leave them out. With respect to the location, we use a structure with on

the lowest level the 16×16 locations, one level up we group sets of 4 locations, resulting in 8×8 aggregated locations, and so on. We perform aggregation on one state variable at a time, achieving an almost exponential decline in the size of the state space. An overview of the aggregation structure is given in Table 3.2, where a '*' corresponds to a state variable included in the aggregation level and a '–' indicates that it is aggregated out.

Table 3.2: Hierarchical aggregation structure

Level	Location	Trailer type	Day of the week	Size of the state space
0	(16×16)	*	*	$256 \times 3 \times 7 = 5376$
1	(8×8)	*	*	$64 \times 3 \times 7 = 1344$
2	(4×4)	*	*	$16 \times 3 \times 7 = 336$
3	(4×4)	–	*	$16 \times 1 \times 7 = 112$
4	(2×2)	–	*	$4 \times 1 \times 7 = 28$
5	–	–	*	$1 \times 1 \times 7 = 7$
6	–	–	–	$1 \times 1 \times 1 = 1$

Computational Results

The results of using the hierarchical aggregation approach are shown in Fig. 3.3, policy $Heps\varepsilon$, with $\varepsilon = 0.25$ and $\varepsilon = 1$. This policy does not require a stepsize, since this is included in the updating equations of the hierarchical aggregation approach. Clearly, the $Heps\varepsilon$ policies converge faster to the optimal values. The policy Heps1 results in only 8% lower profits compare to the optimal policy.

Finally, to gain insight into the long term performance, we show the results using 25,000 iterations, but with fewer replications ($K = 10$), in Fig. 3.4. After 25,000 measurements, the policies Eps1-B and Heps1-B are both within 1% of the optimal performance.

Next, we show the results for the finite horizon case in Fig. 3.5. Here, we both consider the single pass (SP) and the double pass (DP) version of the ADP algorithm. Here, the pure exploitation policy does not benefit from a double pass, simply because with the double pass, the decisions will be even more biased towards states visited before. The same also holds for the ε-greedy policies, since they explore only 5% of the time. However, the hierarchical ε-greedy policies do benefit from the double pass. In addition, the hierarchical ε-greedy policies also benefit from exploration (Heps005). With increasing number of measurements, the hierarchical ε-greedy policies are eventually outperformed by the single pass ε-greedy policies (Heps005-Double 2.56% away from optimum and Eps005-Single 1.56% away from optimum). However, using 25,000 measurements is not representative for a problem

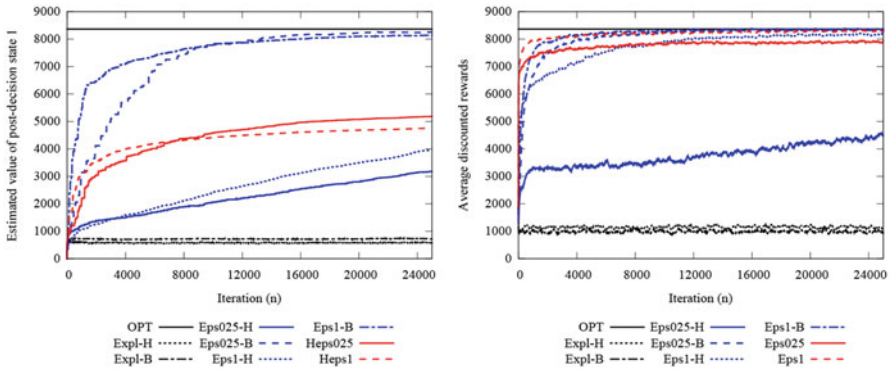


Fig. 3.4: Infinite horizon single-attribute case: resulting estimate $\overline{V}_0^n\left(S_0^{x,n}\right)$ (*left*) and realized rewards (*right*), using $N = 25{,}000$, $M = 10$, $O = 1000$ and $K = 10$. For the rewards resulting from the simulations, the 2500 observations are smoothed using a window of 10

having a state space with size 256. In most ADP applications, the number of iterations would be only a fraction of the size of the state space, say a couple of hundred in this example. Clearly, in these cases the hierarchical aggregation approach performs best. The results after 200 measurements, for various policies, including on-policy and off-policy variations, can be found in Fig. 3.6.

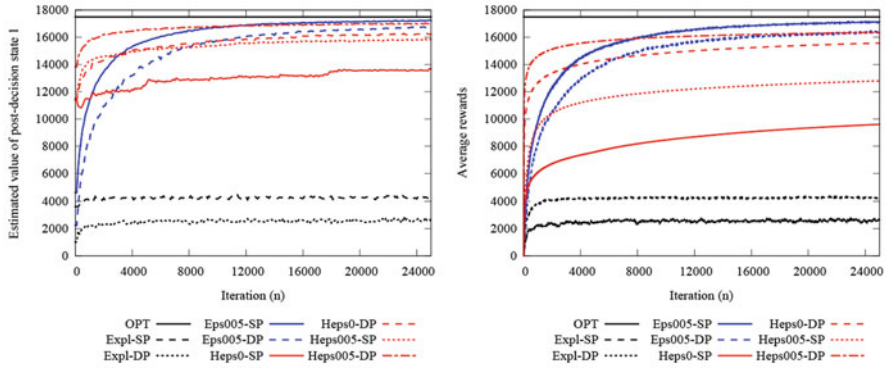


Fig. 3.5: Finite horizon single-attribute case: resulting estimate $\overline{V}_0^n\left(S_0^{x,n}\right)$ (*left*) and realized rewards (*right*), using $N = 25{,}000$, $M = 100$, $O = 100$ and $K = 10$. For the exploitation and ε-greedy policies, the BAKF stepsize is used

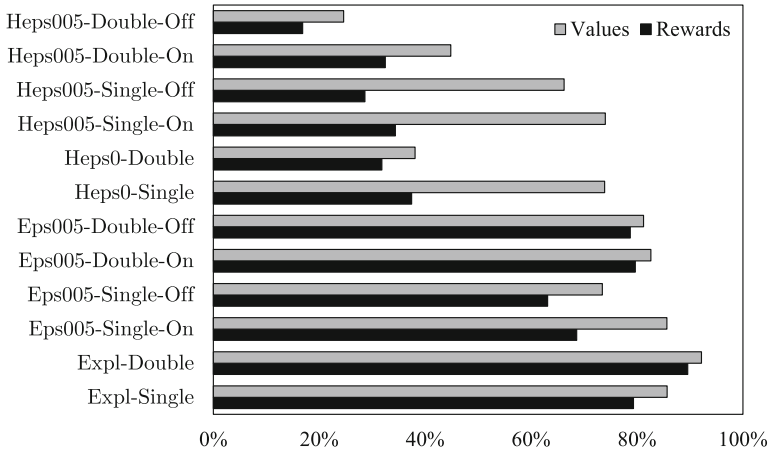


Fig. 3.6: Finite horizon single-attribute case: deviation of the estimate $\overline{V}_0^n\left(S_0^{x,n}\right)$ and realized rewards from the optimal values, using $N = 200$, $O = 100$ and $K = 10$

The results for the multi-attribute case can be found in the Appendix. The results are similar to the those observed for the single-attribute case. One remarkable behavior is that the ε-greedy policy shows an initial decline in performance after which it improves. Again, this is caused by the fact that the decisions are biased towards visiting states that have been measured before, resulting in relatively high direct costs. Once the value of enough states are known, the performance improves. Obviously, this behavior is not visible for the Hierarchical ε-greedy policy since it is able to generalize across states.

3.3 A Freight Consolidation Example

First, we briefly introduce the problem in Sect. 3.3.1 after which we present the MDP model (Sect. 3.3.2) and the ADP approach (Sect. 3.3.3).

3.3.1 Problem Introduction

We now consider another transportation example, with completely different characteristics. We use this example to (1) illustrate the typical use of ADP for resource allocation problems and (2) to illustrate a value function approximation relying on the basis function approach.

We consider the planning problem that arises when a company transports freights from a single origin to different destinations, periodically, using a high

capacity mode. The destinations of these freights are far away and closer among themselves than to the origin of transportation. For this reason, the long-haul is the same in every trip, independent of which freights were consolidated at the origin. However, the last-mile route varies according to the destinations of the freights that were consolidated at the beginning of the long-haul. In addition, there is an alternative, low capacity mode that can be used to transport freights directly from the origin to their destination. Each period, the choice is which freights to allocate in the current period to the high capacity mode, which ones to transport with the low capacity mode, and which ones to postpone to a later period. The costs of the long-haul are fixed, but the last-mile costs depend on the combination of destinations visited. The costs per freight for the low capacity mode are considerably higher than the high capacity mode. The objective of the company is to reduce its total costs over time and to use the long-haul, high capacity mode capacity efficiently. Properly balancing the consolidation and postponement of freights, such that only a few close-by destinations are visited each day, is therefore a challenge for the company but also a necessity for its efficient operation.

We consider a dynamic multi-period long-haul freight consolidation problem where decisions are made on consecutive periods t over a finite horizon $\mathcal{T} = \{0,1,2,\ldots,T^{max}-1\}$. For simplicity, we refer to a period as a day in the remainder of this example. Each freight must be delivered to a given destination d from a group of destinations $\mathcal{D}$ within a given time-window. The time-window of a freight begins at a release-day $r \in \mathcal{R} = \{0,1,2,\ldots,R^{max}\}$ and ends at a due-day $r+k$, where $k \in \mathcal{K} = \{0,1,2,\ldots,K^{max}\}$ defines the length of the time-window. The arrival-day t of a freight is the moment when all its information is known to the planner. Note that r influences how long the freights are known before they can be transported, and thus influences the degree of uncertainty in the decisions.

New freights become available as time progresses. These freights and their characteristics follow a stochastic arrival process. Between two consecutive days, a number of freights f arrive with probability p_f^F, independent of the arrival day. Each freight has destination d with probability p_d^D, release-day r with probability p_r^R, and time-window length k with probability p_k^K, independent of the day and of other freights.

Each day, there is only one long-haul vehicle which transports at most Q freights. Its cost is $C_{\mathcal{D}'}$, where $\mathcal{D}' \subseteq \mathcal{D}$ denotes the subset of destinations visited. There is also an alternative transport mode for each destination d, which can only be used for freights whose due-day is immediate (i.e., $r = k = 0$). The cost of the alternative transport mode is B_d per freight to destination d, and there is no limit on the number of freights that can be transported using this mode.

3.3.2 MDP Model

We subsequently present the following elements of the MDP model: the state (Sect. 3.3.2.1), the decision (Sect. 3.3.2.2), the costs (Sect. 3.3.2.3), the new information and transition function (Sect. 3.3.2.4), and the solution (Sect. 3.3.2.5).

3.3.2.1 State

At each time period t, there are known freights with different characteristics. We define $F_{t,d,r,k}$ as the number of known freights at stage t, whose destination is d, whose release-day is r stages after t, and whose time-window length is k (i.e., its due-day is $r+k$ stages after t). The state of the system at stage t is denoted by S_t and is defined as the vector of all freight variables $F_{t,d,r,k}$, as seen in (3.16).

$$S_t = \left[F_{t,d,r,k}\right]_{\forall d \in \mathcal{D}, r \in \mathcal{R}, k \in \mathcal{K}} \tag{3.16}$$

3.3.2.2 Decision

The decision at each stage is which released freights (i.e., freights with $r = 0$) to consolidate in the long-haul vehicle. We use the integer variable $x_{t,d,k}$ as the number of freights that are consolidated in the long-haul vehicle at stage t, which have destination d and are due k stages after t. We denote the vector of decision variables at stage t as x_t. Due to the time-window of freights, the possible values of these decision variables are state dependent. Thus, the feasible space of decision vector x_t, given a state S_t, is as follows:

$$x_t = \left[x_{t,d,k}\right]_{\forall d \in \mathcal{D}, k \in \mathcal{K}} \tag{3.17a}$$

s.t.

$$\sum_{d \in \mathcal{D}} \sum_{k \in \mathcal{K}} x_{t,d,k} \leq Q, \tag{3.17b}$$

$$0 \leq x_{t,d,k} \leq F_{t,d,0,k} \tag{3.17c}$$

3.3.2.3 Costs

The cost of a decision x_t at a state S_t depends on the destinations visited by the long-haul vehicle and the use of the alternative mode (i.e., $C_{\mathcal{D}'}$ and B_d, respectively). The costs at stage t are given by $C(S_t, x_t)$. From the decision x_t, we derive the combination of terminals that will be visited by the high capacity mode (which determines the high capacity vehicle costs) as well as the number of urgent freights that are not scheduled to be delivered by the high capacity mode (which determines the low capacity vehicle costs).

3.3.2.4 New Information and Transition Function

We introduce a single arrival information variable $\widetilde{F}_{t,d,r,k}$, which represents the freights that arrived from outside the system between stages $t-1$ and t, with

destination d, release-day r, and time-window length k. We denote the vector of all arrival information variables at stage t as W_t, as seen in (3.18).

$$W_t = \left[\widetilde{F}_{t,d,r,k}\right]_{\forall d \in \mathcal{D}, r \in \mathcal{R}, k \in \mathcal{K}} \tag{3.18}$$

The consolidation decision x_t and arrival information W_t have an influence on the transition of the system between stages $t-1$ and t. In addition, the relative time-windows have an influence on the transition between related freight variables. To represent all of these relations, we use the following transition function:

$$S_t = S^M\left(S_{t-1}, x_{t-1}, W_t\right) | t > 0 \tag{3.19a}$$

s.t.

$$F_{t,d,0,k} = F_{t-1,d,0,k+1} - x_{t-1,d,k+1} + F_{t-1,d,1,k} + \widetilde{F}_{t,d,0,k}\ ,\ k < K^{max} \tag{3.19b}$$

$$F_{t,d,0,K^{max}} = F_{t-1,d,1,K^{max}} + \widetilde{F}_{t,d,0,K^{max}} \tag{3.19c}$$

$$F_{t,d,r,k} = F_{t-1,d,r+1,k} + \widetilde{F}_{t,d,r,k}\ ,\ 1 \leq r < R^{max} \tag{3.19d}$$

$$F_{t,d,R^{max},k} = \widetilde{F}_{t,d,R^{max},k} \tag{3.19e}$$

For the transition of the freight variables $F_{t,d,r,k}$ in (3.19a), we distinguish between four cases. First, freights which are already released at stage t (i.e., $r = 0$) and have a time-window length of $k < K^{max}$ are the result of: (1) freights from the previous stage $t-1$ which were already released, had time-window length $k+1$, and were not transported (i.e., $F_{t-1,d,0,k+1} - x_{t-1,d,k+1}$), (2) freights from the previous stage $t-1$ with next-stage release-day (i.e., $r = 1$) and time-window length k (i.e., $F_{t-1,d,1,k}$), and (3) the new (random) arriving freights with the same characteristics (i.e., $\widetilde{F}_{t,d,0,k}$) as seen in (3.19b). Second, freights that are already released at day t and have a time-window length $k = K^{max}$ are the result of freights from the previous stage $t-1$ that had a next day release and the same time-window length (i.e., $F_{t-1,d,1,K^{max}}$), in addition to the freights that arrived between the previous and the current day with the same characteristics (i.e., $\widetilde{F}_{t,d,0,K^{max}}$), as seen in (3.19c). Third, freights which are released at stage t (i.e., $r \geq 1$) are the result of: (1) freights from the previous stage $t-1$ with a release-day $r+1$ and that have the same time-window length k, and (2) the new freights with the same characteristics (i.e., $\widetilde{F}_{t,d,r,k}$), as seen in (3.19d). Fourth, freights which have the maximum release-day (i.e., $r = R^{max}$) are the result only of the new freights with the same characteristics (i.e., $\widetilde{F}_{t,d,R^{max},k}$), as seen in (3.19e).

3.3.2.5 Solution

Again, the formal objective of the model is to find the policy $\pi \in \Pi$ that minimizes the expected costs over the planning horizon, given an initial state S_0, as seen in (3.1). Following Bellman's principle of optimality, the best policy π for the planning horizon can be found solving a set of stochastic recursive equations that

consider the current-stage and expected next-stage costs, as seen in (3.3). We can solve (3.3) plugging in the transition function (3.19a) and specifying the probability $\mathbb{P}(W_{t+1} = \omega)$, which can be found in [9].

Naturally, only very small problem instances can be solved to optimality. The instance we use in this example has the following characteristics. We consider a planning horizon of a working week ($T^{max} = 5$), three destinations, ($|\mathcal{D}| = 3$), one release-day ($R^{max} = 0$), three time-window lengths ($K^{max} = 2$), and at most two freights per day ($|\mathcal{F}| = 2$). The capacity of the long-haul, high capacity vehicle is $Q = 2$. All probabilities and costs are given in the Appendix.

The given problem settings result in an MDP model with 2884 states. For simplicity, we choose to explain more into detail two of these states. The first state, refereed to as "State 1" has only one freight for destination 2 with a time-window length of 2 (i.e. $F_{0,2,0,2} = 1$). The second state, referred to as "State 2", has a total of six freights: one urgent freight for destination 2 and 3, three freights for destination 2 with time-window length 1, and one freight for destination 2 with time-window length 2 (i.e., $F_{0,2,0,0} = F_{0,3,0,0} = 1$, $F_{0,2,0,1} = 3$, $F_{0,2,0,2} = 1$). The optimal costs for State 1 and State 2 are 968.15 and 2619.54, respectively. We choose these two states to show, in the following, the different design challenges arising when applying the ADP algorithm with basis functions to different initial states.

3.3.3 Approximate Dynamic Programming

The ADP approach is presented using a similar setup as with the first example. We subsequently present the post-decision state (Sect. 3.3.3.1), the forward dynamic programming approach (Sect. 3.3.3.2), and the use of value function approximations (Sect. 3.3.3.3).

3.3.3.1 Post-decision State

The post-decision state contains all post-decision freight variables $F^{x,n}_{t,d,r,k}$:

$$S^{x,n}_t = \left[F^{x,n}_{t,d,r,k}\right]_{\forall d\in\mathcal{D}, r\in\mathcal{R}, k\in\mathcal{K}} \tag{3.20}$$

We use the following function $S^{M,x}$ for the transition from the state S^n_t to the post-decision state $S^{x,n}_t$:

$$S^{x,n}_t = S^{M,x}(S^n_t, x^n_t) \tag{3.21a}$$

s.t.

$$F^{x,n}_{t,d,0,k} = F^n_{t,d,0,k+1} - x^n_{t,d,k+1} + F^n_{t,d,1,k}\ ,\ k < K^{max} \tag{3.21b}$$

$$F^{x,n}_{t,d,0,K^{max}} = F_{t-1,d,1,K^{max}} \tag{3.21c}$$

$$F^{x,n}_{t,d,r,k} = F^n_{t,d,r+1,k}\ ,\ 1 \le r < R^{max} \tag{3.21d}$$

This function works in the same way as the MDP transition function defined in (3.19a), with the difference that the new arrival information W_t is not included. An illustration of the transition of states can be found in Fig. 3.7.

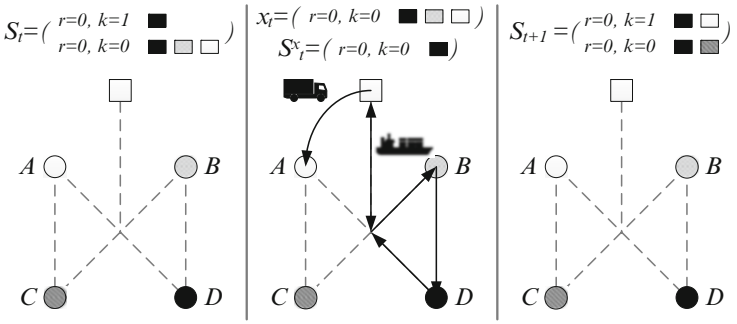


Fig. 3.7: Transition of states in the freight consolidation problem

3.3.3.2 Forward Dynamic Programming

We can directly apply Algorithm 1 using the lookup-table approach. In a minimization problem, initializing the values of the lookup-table with zero will automatically result in an "exploration policy". In such an initialization, a post-decision state that has not been visited before is more attractive (zero downstream costs) than one that has been visited before and has resulted in some costs. In our example, we choose to initialize the values to zero to take advantage of the exploration behavior. Furthermore, we use the harmonic stepsize (see Sect. 3.2.3.2) with $\lambda = 25$ and $\alpha^0 = 0.05$. The ADP runs for a total of 250 iterations, using the double pass approach. The estimated values (i.e., learned costs) for State 1 and State 2 are 1153.85 and 2814.74, respectively. These values are 19% and 7% higher then the optimal MDP costs, respectively. The average costs for State 1 and State 2 (obtained through a simulation of the policy resulting from the learned values) are 1550.65 and 2852.75, respectively. These average costs are 19% and 9% higher then the optimal MDP costs, respectively. In the following, we elaborate on how the performance of the value function approximation can be improved through the use of basis functions.

3.3.3.3 Value Function Approximation

For this example, we introduce a frequently used approximation strategy using basis functions. An underlying assumption in using basis functions is that particular features, or quantitative characteristics, of a (post-decision) state can be identified, that explain, to some extent, what the value of that post-decision state is. In our problem,

features such as the number of urgent freights, the number of released freights that are not urgent, and the number of freights that have not been released for transport, can explain part of the value of a post-decision state. Basis functions are then created for each individual feature to quantify the impact of the feature on the value function.

We define a set of features $\mathcal{A}$ for which the value of each feature $a \in \mathcal{A}$ of a post-decision state $S_t^{x,n}$ is obtained using a basis function $\phi_a(S_t^{x,n})$. We assume the approximated next-stage value of a post-decision state can be expressed by a weighted linear combination of the features, using the weights θ_a^n for each feature $a \in \mathcal{A}$, as follows:

$$\overline{V}_t^{x,n}\left(S_t^{x,n}\right) = \sum_{a \in \mathcal{A}} \theta_a^n \phi_a\left(S_t^{x,n}\right) \tag{3.22}$$

The weight θ_a^n is updated recursively in each iteration n. Note that (3.22) is a linear approximation, as it is linear in its parameters. The basis functions themselves can be nonlinear [10].

The use of features and weights for the approximating the value function $\overline{V}_t^n(S_t^{x,n})$ is comparable to the use of regression models for fitting data to a (linear) function. In that sense, the independent variables of the regression model would be the features of the post-decision state and the dependent variable would be the value of the post-decision state. However, in contrast to regression models, the data in our ADP is generated iteratively inside an algorithm and not all at once. Therefore, the updating process U^V for the approximating function in (3.22) cannot be based only on solving systems of equations as in traditional regression models.

Several methods are available to "fine-tune" the weights θ_a^n for each feature $a \in \mathcal{A}$ after each iteration. An effective approach is the recursive least squares method, which is a technique to compute the solution to a linear least squares problem [10]. Two types of recursive least squares methods are available. The least squares method for *nonstationary* data provides the opportunity to put increased weight on more recent observations, whereas the least squares method for *stationary* data puts equal weight on each observation. For the purpose of learning the weights within an ADP algorithm, the recursive least squares method for nonstationary data is more appropriate. The method for updating the value function approximations with the recursive least squares method for nonstationary data is explained in detail in [10]. Nevertheless, the equations used in this method are given below.

The weights θ_a^n, for all $a \in \mathcal{A}$, are updated each iteration (n is the iteration counter) by

$$\theta_a^n = \theta_a^{n-1} - H_n \phi_a\left(S_t^{x,n}\right)\left(\overline{V}_{t-1}^{n-1}\left(S_{t-1}^{x,n}\right) - \hat{v}_t^n\right),$$

where H^n is a matrix computed using

$$H^n = \frac{1}{\gamma^n} B^{n-1},$$

and where B^{n-1} is an $|\mathcal{A}|$ by $|\mathcal{A}|$ matrix that is updated recursively using

$$B^n = \frac{1}{\alpha^n}\left(B^{n-1} - \frac{1}{\gamma^n}\left(B^{n-1}\phi\left(S_t^{x,n}\right)\left(\phi\left(S_t^{x,n}\right)\right)^T B^{n-1}\right)\right).$$

The expression for γ^n is given by

$$\gamma^n = \alpha^n + \phi\left(S_t^{x,n}\right)^T B^{n-1}\phi\left(S_t^{x,n}\right).$$

B^n is initialized by using $B^0 = \varepsilon I$, where I is the identity matrix and ε is a small constant. This initialization works well when the number of observations is large [10]. The parameter α^n determines the weight on prior observations of the value. Setting α^n equal to 1 for each n would set equal weight on each observation, and implies that the least squares method for stationary data is being used. Setting α^n to values between 0 and 1 decreases the weight on prior observations (lower α^n means lower weight). We define the parameter α^n by

$$\alpha^n = \begin{cases} 1 & \text{, stationary} \\ 1 - \frac{\delta}{n} & \text{, nonstationary} \end{cases} \tag{3.23}$$

where $1 - \frac{\delta}{n}$, with $\delta = 0.5$, is a function to determine α_n that works well in our experiments.

For this example, there are a number of possible features, such as:

1. Each state variable: number of freights with specific attributes.
2. Per destination, the number of freights that are not yet released for transport (i.e., future freights).
3. Per destination, the number of freights that are released for transport and whose due-day is not immediate (i.e., may-go freights).
4. Per destination, a binary indicator to denote the presence of urgent freights (i.e., must-visit destination).
5. For each state variable, some power function (e.g., a^2) to represent non-linear components in costs.

We test various combinations of the features mentioned above and name them Value Function Approximations (VFA) 1, 2 and 3 (see the Appendix for their settings).

Computational Results

Intuitively, the postponed freights and their characteristics influence the future costs of a decision. However, measuring how these characteristics influence the costs, and thus determining which VFA is the best one to use, is challenging. For small instances of the problem, one option to determine the best set of features to use is to perform a linear regression between the optimal values of all states of the MDP and the basis functions corresponding to each set of features, and choose the set with the highest coefficient of determination R^2. Another option, applicable to medium sized instances of the problem is to calculate the average costs of a subset of states, using each set of features, in three steps: (1) run the ADP algorithm for a subset of all states, using the different sets of features, (2) simulate the resulting policies for

a number of iterations, and (3) repeat the first and second step a number of replications. In the case of this small example, we perform the two options and present the results in Table 3.3 and Fig. 3.8. For the second option, we simulate the resulting policies of all states and show the average difference between the average costs of the simulation and the optimum value of each state. Although the differences among the sets of features in both tests are small, we note that considering them one at a time would lead to different conclusions. With the coefficient of determination of the linear regression, VFA 2 would be selected as the best. However, with the average costs approach, VFA 3 would be selected. In addition to having the lowest average difference, VFA 3 also has the smallest variance of the three sets of features.

Table 3.3: Performance of the different VFAs

Test indicator	Lookup-table	VFA 1	VFA 2	VFA 3
R^2	–	0.8897	0.8915	0.8897
Average difference (%)	7.50	2.67	2.45	2.36

The two aforementioned tests of the various combinations of features consider all states of the MDP. A third approach to decide which basis functions to use, which is applicable to large instances, is to pick some initial state (or multiple initial states) and compare (1) the values learned by the ADP algorithm using various

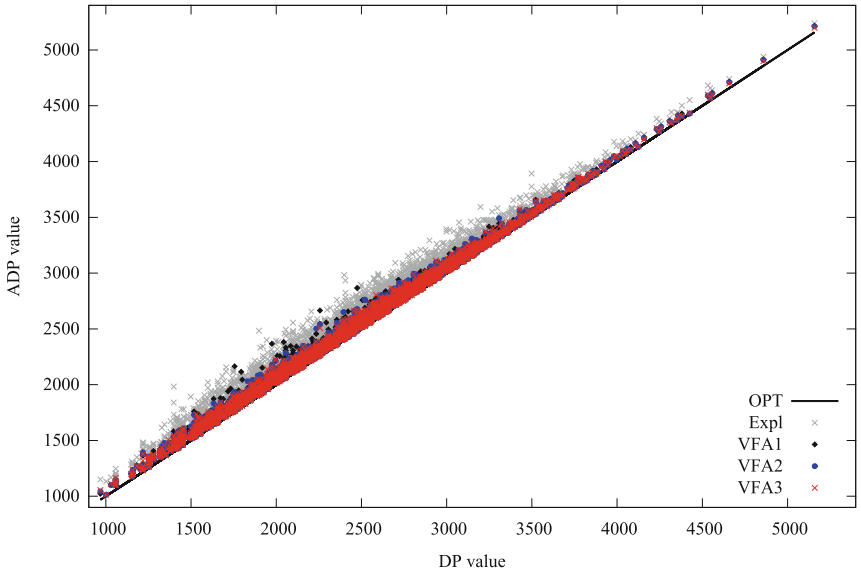


Fig. 3.8: Average performance of the ADP algorithm for the different VFAs compared to the optimal values for all states, using $N = 250$, $M = 250$, $O = 100$, and $K = 10$

sets of features and (2) the resulting performance of the policy resulting from these values. We perform this approach for the two states introduced before, and show the results in Fig. 3.9. Since we use a small problem instance, we also show the optimal value in the figures, as well as the lookup-table approach mentioned earlier in this section (denoted by "Expl"). For all sets of features (VFAs), the ADP algorithm runs for 250 iterations using a double pass approach. In addition to the tests of each VFA, we also test each VFA with an ε-greedy approach ($\varepsilon = 0.05$), and denote these on the graphs by "VFAeps", since this approach yielded good results in our example.

For State 1 in Fig. 3.9, we observe some differences among the VFAs estimated (learned) value and the average costs (performance) of the resulting policy. These differences can lead to choosing different sets of features. On the one hand, the differences among the learned values of the three sets of features indicate that VFA1eps is the best. On the other hand, there are no clear differences among the average costs of all VFAeps, indicating that the three sets perform equally well when using the ε-greedy approach (and all better than all no-ε VFAs). Furthermore, we observe that the ε-greedy approach improves the estimated values in VFA1 and VFA2 (i.e., VFAeps $\leq$ VFA), but not in VFA3. In the case of the average costs, the ε-greedy approach improves all VFAs in a way that their performance is almost the same.

The results for State 1 can lead to the conclusion that a proper tuning of the exploration/exploitation tradeoff (e.g., via the ε-greedy) can have a larger impact on the performance than the set of features chosen. However, an explanation on why this is the case for this state has to do with the state and the sets of features themselves. State 1 is an almost empty state (i.e., only one freight), which means most basis functions of the three sets we test return zero. Remind that the updating algorithm can only determine how significant the weight of a basis function is as long as it observes it. When only one basis function is observed, and this basis function behaves similarly in all sets of features, the updating algorithm will assign similar weights and thus the resulting policies will be approximately the same.

For State 2 in Fig. 3.9, we observe significant differences among the estimated values of the VFAs, but not among the average costs of the resulting policies. Clearly, VFA2 and VFA3eps have the best estimated value, and VFA3 the worst (even worse than the lookup-table approach). However, when looking at the average costs, the policy from all three VFAs (without the ε-greedy approach) seem to achieve the same costs, between 1 and 2% away from the optimal costs. Moreover, the second-best learning set of features (VFA3eps) is now performing second-worst of all seven value function approximation methods tested. This indicates that having good estimates of the value of states do not necessarily result in a good performance.

When looking at all four figures, we can conclude that deciding on which set of features to use requires careful design and testing, and that the quality of the chosen set of features (basis functions) is heavily problem/state dependent. An explanation on this situation has to do with two characteristics of how the basis functions approximate the future costs. First, all weights of the basis functions, which determine the output policy of the ADP algorithm, can only be updated (i.e., improved)

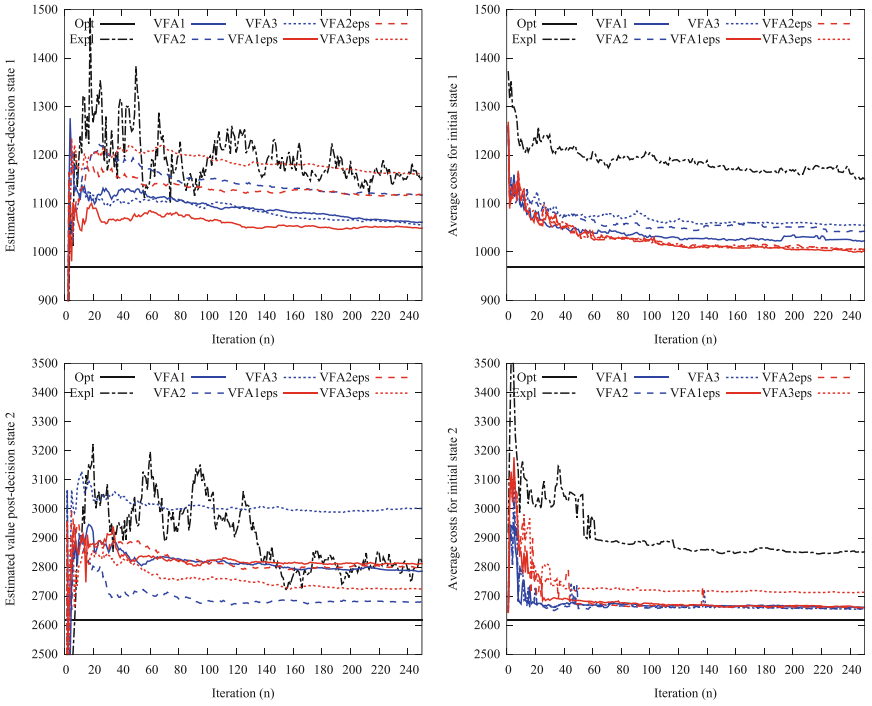


Fig. 3.9: Learned values (*left*) and average cost performance (*right*) of the ADP algorithm for the different VFAs for State 1 (*top*) and State 2 (*bottom*), using $N = 250$, $M = 1$, $O = 100$, and $K = 10$

as long as the basis functions are non-zero. In State 2, which contains many basis functions with a non-zero value, the performance of all VFAs is significantly better than in State 1, which contains mostly basis functions with a value of zero. On average, all six VFAs achieve 2% higher-than-optimal costs in State 2, while they achieve 6% higher-than-optimal costs in State 1. Second, the magnitude with which the weight is updated depends on how much the value of the basis function varies among the different iterations of the ADP algorithm. These might lead to poorly estimating the value itself. In State 2, the difference between the best and worst learning VFA is larger than in State 1. In this example problem, 1.2 freights arrive on average per day (with at most two freights). This means that State 1 is a state one can expect on average whereas State 2 is an exceptionally busy state. Additionally, with the short horizon considered in this problem, the initial conditions (state) can have a large impact on the optimal costs. Thus, problem/state characteristics must be considered when using the basis functions approach.

Besides the need for an evaluation methodology, the observed performance differences between different initial states also gives rise to new VFA designs that use basis functions. For example, using aggregated designs based on categorization of

states can prevent basis function values of zero and can reduce the variation among basis function values. Designing such a VFA with the right set of features is both an art and a science. With creativity about potential causes of future costs, as well as their limits within a problem, efficient and accurate designs can be developed. With structured evaluation methodologies (e.g., regression analysis, design of experiment techniques, statistical control methods), these designs can be tested and further improved to tune the ADP algorithm to a specific problem.

For further reading on this problem, we refer to [9]. In addition, we refer to [18] for a similar ADP approach on a completely different transportation problem.

3.4 A Healthcare Example

In this third and final example, we repeat the same steps as with the previous two examples, with the difference that we only focus on the modeling part. We omit the experimental results of the MDP and ADP model, for which we refer to [6].

3.4.1 Problem Introduction

The problem concerns tactical planning in a hospital, which involves the allocation of resource capacities and the development of patient admission plans. More concretely, tactical plans distribute a doctor's time (resource capacity) over various activities and control the number of patients that should be treated at each care stage (e.g., surgery). The objective is to achieve equitable access and treatment duration for patients. Each patient needs a set of consecutive care stages, which we denote as a care process. Patients are on a waiting list at each care stage in their care process, and the time spent on this waiting list is called access time. Fluctuations in patient arrivals and resource availabilities result in varying access times for patients at each stage in their care process, and for hospitals, this results in varying resource utilization and service levels. To mitigate and address these variations, tactical planning of hospital resources is required.

The planning horizon is discretized in consecutive time periods $\mathcal{T} = \{1, 2, \ldots, T\}$. We include a set of resource types $\mathcal{R} = \{1, 2, \ldots, R\}$ and a set of patient queues $\mathcal{J} = \{1, 2, \ldots, J\}$. We define $\mathcal{J}^r \subseteq \mathcal{J}$ as the subset of queues that require capacity of resource $r \in \mathcal{R}$. Each queue $j \in \mathcal{J}$ requires a given amount of time units from one or more resources $r \in \mathcal{R}$, given by $s_{j,r}$, and different queues may require the same resource. The number of patients that can be served by resource $r \in \mathcal{R}$ is limited by the available resource capacity $\eta_{r,t}$ in time period $t \in \mathcal{T}$. The resource capacity $\eta_{r,t}$ is given in the same time unit as $s_{j,r}$.

After being treated at a queue $j \in \mathcal{J}$, patients either leave the system or join another queue. To model these transitions, we introduce $q_{j,i}$, which denotes the

fraction of patients that will join queue $i \in \mathcal{J}$ after being treated in queue $j \in \mathcal{J}$. To capture arrivals to and exits from outside the "hospital system", we introduce the element 0 (note that the set $\mathcal{J}$ carries no 0-th element by definition). The value $q_{j,0} = 1 - \sum_{i \in \mathcal{J}} q_{j,i}$ denotes the fraction of patients that leave the system after being treated at queue $j \in \mathcal{J}$.

In addition to demand originating from the treatment of patients at other queues within the system, demand may also arrive to a queue from outside the system. The number of patients arriving from outside the system to queue $j \in \mathcal{J}$ at time $t \in \mathcal{T}$ is given by $\lambda_{j,t}$, and the total number of arrivals to the system is given by $\lambda_{0,t}$.

Patients are transferred between the different queues according to transition probabilities $q_{j,i}, \forall j, i \in \mathcal{J}$ independent of their preceding stages, independent of the state of the network and independent of the other patients. Patients arrive at each queue from outside the system according to a Poisson process with rate $\lambda_{j,t}, \forall j \in \mathcal{J}, t \in \mathcal{T}$. The external arrival process at each queue $j \in \mathcal{J}$ in time period $t \in \mathcal{T}$ is independent of the external arrival process at other queues and other time periods. Since all arrival processes are independent, we obtain $\lambda_{0,t} = \sum_{j=1}^{J} \lambda_{j,t}, \forall t \in \mathcal{T}$. We introduce $\mathcal{U} = \{0, 1, 2, \ldots, U\}$ to represent the set of time periods patients can be waiting, i.e., if we decide not to threat a patient that already waited for U time periods, we assume his/her waiting time remains U time periods.

3.4.2 MDP Model

We subsequently present the following elements of the MDP model: the state (Sect. 3.4.2.1), the decision (Sect. 3.4.2.2), the costs (Sect. 3.4.2.3), the new information and transition function (Sect. 3.4.2.4), and the solution (Sect. 3.4.2.5).

3.4.2.1 State

We introduce $S_{t,j,u}$ as the number of patients in queue $j \in \mathcal{J}$ at time $t \in \mathcal{T}$ with a waiting time of $u \in \mathcal{U}$. The state of the system at time period t can be written as $S_t = (S_{t,j,u})_{j \in \mathcal{J}, u \in \mathcal{U}}$.

3.4.2.2 Decision

The decision $x_{t,j,u}$ is how many patients to treat in queue $j \in \mathcal{J}$ at time $t \in \mathcal{T}$, with a waiting time of $u \in \mathcal{U}$. This decision needs to be made for all queues and waiting times, represented by $x_t = (x_{t,j,u})_{j \in \mathcal{J}, u \in \mathcal{U}}$. The set $\mathcal{X}_t$ of feasible decisions at time t is given by

$$\begin{aligned}\mathcal{X}_t = \{\, x_t \mid \\ & x_{t,i,u} \leq S_{t,i,u}, && \forall i \in \mathcal{J}, t \in \mathcal{T}, u \in \mathcal{U} \\ & \textstyle\sum_{j \in \mathcal{J}^r} s_{j,r} \sum_{u \in \mathcal{U}} x_{t,j,u} \leq \eta_{r,t}, && \forall r \in \mathcal{R}, t \in \mathcal{T} \\ & x_{t,j,u} \in \mathbb{Z}_+ && \forall i \in \mathcal{J}, t \in \mathcal{T}, u \in \mathcal{U}\}.\end{aligned} \tag{3.24}$$

As given in (3.24), the set of feasible decisions in time period t is constrained by the state S_t and the available resource capacity $\eta_{r,t}$ for each resource type $r \in \mathcal{R}$.

3.4.2.3 Costs

The cost function $C_t(S_t, x_t)$ related to our current state S_t and decision x_t is set-up to control the waiting time per stage in the care process, so per individual queue ($j \in \mathcal{J}$). We choose the following cost function, which is based on the number of patients for which we decide to wait at least one time unit longer

$$C_t(S_t, x_t) = \sum_{j \in \mathcal{J}} \sum_{u \in \mathcal{U}} c_{j,u} \left(S_{t,j,u} - x_{t,j,u}\right), \qquad \forall t \in \mathcal{T}. \tag{3.25}$$

In general, higher $u \in \mathcal{U}$ will have higher costs as it means a patient has a longer total waiting time.

3.4.2.4 New Information and Transition Function

The vector W_t containing the new information, consists of new patient arrivals and outcomes for transitions between queues. We distinguish between *exogenous* and *endogenous* information in $W_t = \left(\widehat{S}^e_t, \widehat{S}^o_t(x_{t-1})\right), \forall t \in \mathcal{T}$, where the exogenous $\widehat{S}^e_t = \left(\widehat{S}^e_{t,j}\right)_{\forall j \in \mathcal{J}}$ represents the patient arrivals from outside the system, and the endogenous $\widehat{S}^o_t(x_{t-1}) = \left(\widehat{S}^o_{t,j,i}(x_{t-1})\right)_{\forall i,j \in \mathcal{J}}$ represents the patient transitions to other queues as a function of the decision vector x_{t-1}. $\widehat{S}^o_{t,j,i}(x_{t-1})$ gives the number of patients transferring from queue $j \in \mathcal{J}$ to queue $i \in \mathcal{J}$ at time $t \in \mathcal{T}$, depending on the decision vector x_{t-1}.

We use the following transition function:

$$S_t = S^M(S_{t-1}, x_{t-1}, W_t), \tag{3.26}$$

where

$$S_{t,j,0} = \widehat{S}^e_{t,j} + \sum_{i \in \mathcal{J}} \widehat{S}^o_{t,i,j}(x_{t-1,i}), \quad \forall j \in \mathcal{J}, t \in \mathcal{T}, \tag{3.27}$$

$$S_{t,j,U} = \sum_{u=U-1}^{U} \left(S_{t-1,j,u} - x_{t-1,j,u}\right), \quad \forall j \in \mathcal{J}, t \in \mathcal{T}, \tag{3.28}$$

$$S_{t,j,u} = S_{t-1,j,u-1} - x_{t-1,j,u-1}, \quad \forall j \in \mathcal{J}, t \in \mathcal{T}, u \in \mathcal{U} \setminus \{0, U\}, \tag{3.29}$$

are constraints to ensure that the waiting list variables are consistently calculated. Constraint (3.27) determines the number of patients entering a queue. Constraint (3.28) updates the waiting list for the longest waiting patients per queue. The state $S_{t,j,U}$, for all $t \in \mathcal{T}$ and $j \in \mathcal{J}$, holds all patients that have been waiting U time periods and longer. Constraint (3.29) updates the waiting list variables at each time period for all $u \in \mathcal{U}$ that are not covered by the first two constraints. All arrivals in time period $t \in \mathcal{T}$ to queue $j \in \mathcal{J}$ from outside the system ($\widehat{S}^e_{t,j}$) and from internal transitions ($\sum_{i \in \mathcal{J}} \widehat{S}^o_{t,i,j}(x_{t-1,i})$) are combined in (3.27).

3.4.2.5 Solution

Again, the formal objective of the model is to find the policy $\pi \in \Pi$ that minimizes the expected costs over the planning horizon, given an initial state S_0, as seen in (3.1). The exact DP-problem is restricted by limiting the number of patients that can be waiting in each queue to a given maximum. To illustrate the size of the state space for our problem, suppose that $\hat{M}$ gives the maximum number of patients per queue and per number of time periods waiting. The number of states is then given by $\hat{M}^{(|\mathcal{J}|\cdot|\mathcal{U}|)}$. We can solve (3.4) plugging in the transition function (3.26) and specifying the probability $\mathbb{P}(W_{t+1} = \omega)$, which can be found in [6].

3.4.3 Approximate Dynamic Programming

The ADP approach is presented using a similar setup as used in the previous examples. We subsequently present the post-decision state (Sect. 3.4.3.1), the forward dynamic programming approach (Sect. 3.4.3.2), and the use of value function approximations (Sect. 3.4.3.3).

3.4.3.1 Post-decision State

The post-decision state $S^{x,n}_t$ represents the expected results of the decision x_t taken in state S^n_t. More specifically, we subtract the number $x_{t,j,u}$ of patients we decided to treat and use the expected patient transitions $q_{i,j}$ to determine the next location for each of the patients we decided to treat.

The transitions take place as follows. In addition to the transition function (3.26), which gives the transition from the state S^n_t to the state S^n_{t+1}, we introduce a transition function $S^{M,x}(S^n_t, x_t)$, which gives the transition from the state S^n_t to the post-decision state $S^{x,n}_t$. This function is given by:

$$S^{x,n}_t = S^{M,x}(S^n_t, x_t), \tag{3.30}$$

with

$$S^{x,n}_{t,j,0} = \sum_{i\in\mathcal{J}}\sum_{u\in\mathcal{U}} q_{i,j}x_{t,i,u} \qquad \forall j\in\mathcal{J}, t\in\mathcal{T} \tag{3.31}$$

$$S^{x,n}_{t,j,U} = \sum_{u=U-1}^{U} (S_{t,j,u} - x_{t,j,u}) \qquad \forall j\in\mathcal{J}, t\in\mathcal{T} \tag{3.32}$$

$$S^{x,n}_{t,j,u} = S_{t,j,u-1} - x_{t,j,u-1} \qquad \forall j\in\mathcal{J}, t\in\mathcal{T}, u\in\mathcal{U}\setminus\{0,U\}. \tag{3.33}$$

The transition function (3.30) closely resembles (3.26), except that the external arrivals to the system and the final realization of the patient transitions $q_{i,j}$ are not included.

Due to the patient transfer probabilities, the transition function (3.30) may result in non-integer values for the post-decision state. We do not round these values as the post-decision state is only used to provide a value estimate from a particular combination of a state and a decision. Hence, the post-decision state is only used as an 'estimate' of the future state. The post-decision state will not be used to compute the transition from state S_t to state S_{t+1}. Within the ADP algorithm, we use the original transition function (3.26) to compute the state in the next time period. As a result, the post-decision state will not cause any state to become non-integer.

The actual realizations of new patient arrivals and patient transitions in a time period will be incorporated in the transition to the state in the next time period. An illustration of the transition of states can be found in Fig. 3.10.

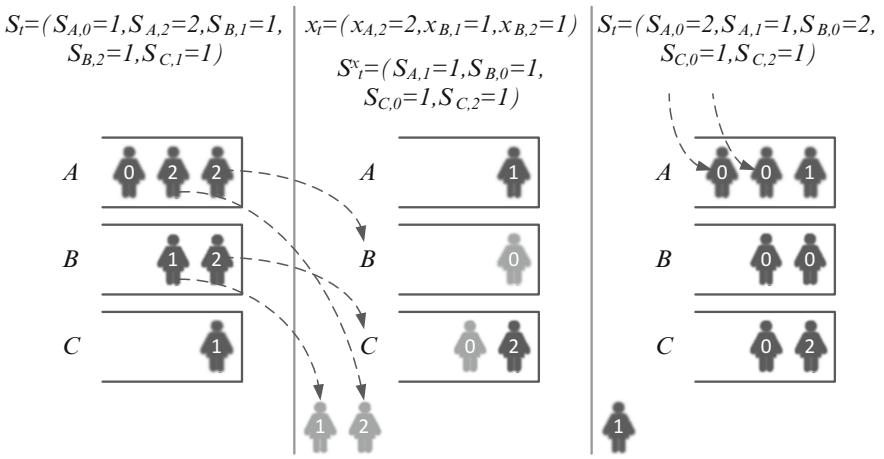


Fig. 3.10: Transition of states in the healthcare problem

3.4.3.2 Forward Dynamic Programming

We use a the same forward dynamic programming approach as presented in Sect. 3.3.3.2.

3.4.3.3 Value Function Approximation

Again, the challenge is to design a proper approximation for the 'future' costs $\overline{V}_t^n\left(S_t^{x,n}\right)$ that is computationally tractable and provides a good approximation of the actual values. Similar to the previous example, we make use of basis functions.

For our application, we make the assumption that the properties of each queue are independent from the properties of the other queues, so that we can define basis functions for each individual queue that describe important properties of that queue. For the basis functions, we choose to use the features 'The number of patients in queue j that are waiting for u periods'. These features result in the following basis functions that will be used in the ADP algorithm: $S_{t,j,u}, \forall j \in \mathcal{J}, \forall u \in \mathcal{U}, t = 1$. The basis functions explain a large part of the variance in the computed values with the exact DP approach ($R^2 = 0.954$) and they can be straightforwardly obtained from the post decision state. The weights θ^n in the value function approximations are initialized to $\theta^0 = 1$ for all time periods, and we use the matrix $B^0 = \varepsilon I$ as explained before. We use the double pass version of the ADP algorithm and determine the stepsize α using nonstationary least squares with $\delta = 0.99$. All other settings can be found in [6].

In case there is no independent constant in the set of predictors $\mathcal{F}$ in a linear regression model, the model is forced to go through the origin (all dependent and independent variables should be zero at that point). This may cause a bias in the predictors. To prevent this bias, we add a constant term as one of the elements in $\mathcal{F}$. The feature weight θ_f^n may vary, but the feature value $\phi_f\left(S_t^{x,n}\right)$ of this constant is always 1, independent of the state $S_t^{x,n}$.

We have calculated the ADP-algorithm for 5000 random states and found that the values found with the ADP algorithm and the value from the exact DP solution converge. For these 5000 random states, there is an average deviation between the value approximated with the ADP algorithm and the value calculated with the exact DP approach of 2.51%, with a standard deviation of 2.90%, after 500 iterations. This means the ADP algorithm finds slightly larger values on average than the exact DP approach. This may be caused by the truncated state space, as explained before. The calculation time of the ADP algorithm is significantly lower than the calculation of the exact DP solution. Obtaining the DP solution requires over 120 h. Calculating the ADP solution for a given initial state (with $N = 500$) takes on average only 0.439 s. For a complete analysis of this approach, we refer to [6].

3.5 What's More

ADP is a versatile framework that is studied and applied to a growing number of diverse problems. Naturally, diverse problems require a deeper focus on diverse aspects of ADP. In some problems, a correct design of a value function is of most importance for approximating the optimal solution of an MDP, whereas in others, the right tuning of exploration vs exploitation parameters has a higher impact.

Furthermore, in some practical applications, approximating a restricted policy might be better than approximating the values. In this section, we briefly touch upon some of these aspects. We subsequently present various options for policies (Sect. 3.5.1), value function approximations (Sect. 3.5.2), and handling the exploration vs exploitation tradeoff (Sect. 3.5.3).

3.5.1 Policies

In this chapter, we used policies based on value function approximations. There are many other options, like the use of myopic policies, look-ahead policies (rolling horizon procedures that optimize over multiple time periods into the future), and policy function approximations (analytic functions that return an action for each state). And, of course it is possible to use hybrids.

The approach used in this chapter relies on approximate value iteration. Another option is approximate policy iteration. In this strategy, we simulate a policy a number of 'inner' iterations over some horizon. During these inner iterations, we fix the policy (typically by fixing the value function approximation) to obtain a better estimate of the value of being in a state. We refer to [10, Chaps. 9 and 10] for more information on this. Finally, besides value iteration and policy iteration, we can also use the linear programming method. This method—that for MDPs suffers from the curse of dimensionality since we need a decision variable for each state, and a constraint for each state-action pair—receives attention in the ADP community due to [2], where ADP concepts are applied to this method, incorporating value function approximations into the linear program and sampling of the constraints.

More information on different types of policies, and ADP modeling in general, can be found in [11–13] and, using examples from transportation and logistics, [14]. In these works, also the relationship between (approximate) dynamic programming and other techniques as stochastic programming, simulation, and stochastic search, is discussed. A comparison of different ADP techniques, using an energy storage problem, is given in [7].

3.5.2 Value Function Approximations

Value function approximations can be divided into lookup tables (including state space aggregation, hierarchical aggregation, and representatives), parametric models (basis functions, piece-wise linear functions, and neural networks), and nonparametric models (kernel regression and support vector machines). In general, approximating value functions involves the application of statistical methods to estimate the value of being in a state. One specific technique, applicable to the approaches considered in this chapter, involves the selection of basis functions, see [17]. For an overview of statistical learning techniques that can be used in this setting, we refer to [5].

The estimation of value function approximations, however, involves much more than the application of statistical methods. A unique setting of ADP is that value function approximations have to be estimated recursively. Especially during the initial iterations, our decisions are influenced by the initialized values and might be strongly biased by the number of measurements taken from the different states. In addition, testing the performance of a VFA design is a task that requires diverse methods as well, as we illustrated in this chapter.

3.5.3 Exploration vs Exploitation

The exploration vs exploitation tradeoff involves the decisions whether to explore states just to learn their value or to visit the states that appear to be the best. For this purpose, we introduced the ε-greedy policy. The disadvantage of this policy is that the learning rate remains constant and there is no focus on certain areas of the state space. A 'good' policy supports the balance between the estimated value of states and the uncertainty about these values. This problem received considerable attention by the machine learning community (problems related to the Multi-armed Bandit Problem, Ranking and Selection, Bayesian Global Optimization, etc.). These techniques can also be used within ADP. An example can be found in [15], where a new exploration strategy is proposed based on the knowledge gradient concept, in which the uncertainty about the value function is explicitly expressed using a Bayesian model with correlated beliefs. The hierarchical aggregation method described in this chapter has also been extended with a Bayesian belief model in [8]. We incorporated this Hierarchical Knowledge Gradient method within ADP, using a similar approach as presented in [15], and tested it on the nomadic trucker problem from Sect. 3.2. For all iterations $10 \leq n \leq 250$, this exploration technique consistently outperforms the other policies show in Fig. 3.3, both with respect to the learned value functions and the resulting performance. For a more in-depth discussion of strategies to balance exploration and exploitation, we refer to [16], [10, Chap. 10], and [13].

Appendix

Nomadic Trucker Settings

Transportation takes place in a square area of 1000×1000 miles. The locations lie on a 16×16 Euclidean grid placed on this area, where each location $i \in \mathcal{L}$ is described by an (x_i, y_i)-coordinate. The first location has coordinate $(0,0)$ and the last location (location 256) has coordinate $(1000, 1000)$. The minimum distance between two locations is $1000/15$.

For each location $i \in \mathcal{L}$, there is a number $0 \leq b_i \leq 1$ representing the probability that a load originating at location i will appear at a given time step. The probability that, on a given day of the week d, a load from i to j will appear is given by $p_{ij}^d = p_d b_i (1 - b_j)$, where p_d gives the probability of loads appearing on a given day of the week d. The origin probabilities b_i are given by

$$b_i = \rho \left(1 - \frac{f(x_i, y_i) - f^{min}}{f^{max} - f^{min}} \right), \tag{3.34}$$

where ρ gives the arrival intensity of loads, and $f(x_i, y_i)$ is the Six-hump camel back function given by $f(x_i, y_i) = 4x_i^2 - 2.1x_i^4 + \frac{1}{3}x_i^6 + x_i y_i - 4y_i^2 + 4y_i^4$ on the domain $(x_i, y_i) \in [-1.5, 2] \times [-1, 1]$. The highest value is achieved at coordinate $(2, 1)$, with a value of ≈ 5.73, which we reduce to 5 to create a somewhat smoother function (still the second highest value is ≈ 4.72). Next, the values $f(x_i, y_i)$ are scaled to the domain $(x_i, y_i) \in [0, 0] \times [1000, 1000]$. The values $f^{min} = min_{i \in \mathcal{L}} f(x_i, y_i) \approx -1.03$ and $f^{max} = max_{i \in \mathcal{L}} f(x_i, y_i) = 5$ are used to scale $f(x_i, y_i)$ between $[0, 1]$. An impression of the resulting origin probabilities B_i is given in Fig. 3.11.

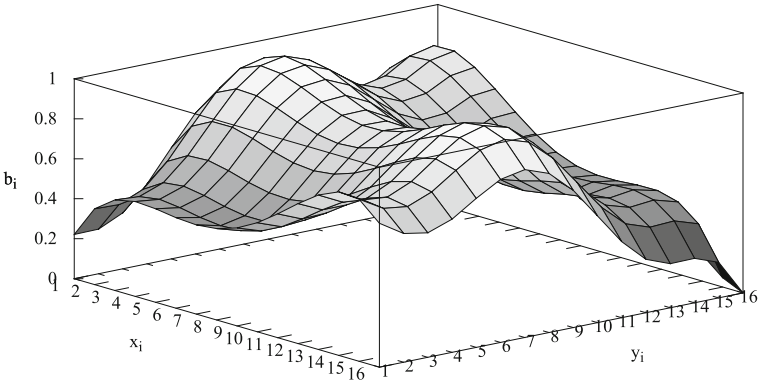


Fig. 3.11: Origin probabilities for the 256 locations

We set $\rho = 1$, which corresponds with an expectation of approximately 93.14 outgoing loads from the most popular origin location on the busiest day of the week. We use a load probability distribution $p^d = (1, 0.8, 0.6, 0.7, 0.9, 0.2, 0.1)$, for d from Monday till Sunday, which represents the situation in which loads are more likely to appear during the beginning of the week (Mondays) and towards the end (Fridays).

The results for the infinite horizon multi-attribute version of the nomadic trucker problem can be found below (Fig. 3.12).

Freight Consolidation Settings

Either one or two freights arrive each period (i.e., $\mathcal{F} = \{1, 2\}$), with probability $p_f^F = (0.8, 0.2)$ for $f \in \mathcal{F}$. Each freight that arrives has destination $d \in \mathcal{D} = \{1, 2, 3\}$

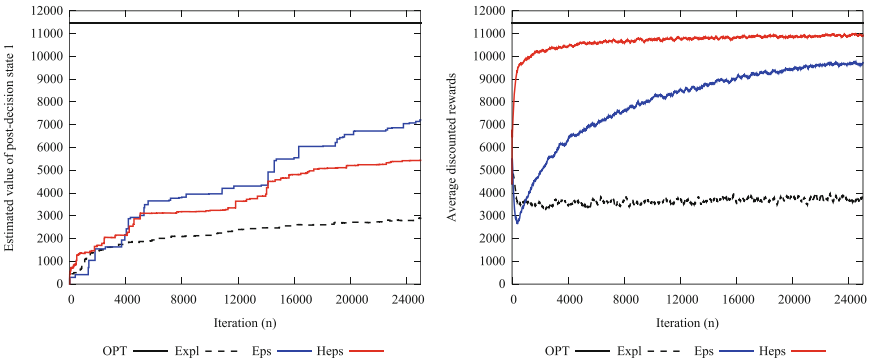


Fig. 3.12: Infinite horizon multi-attribute case: resulting estimate $\overline{V}_0^n\left(S_0^{x,n}\right)$ (*left*) and realized rewards (*right*), using $N = 25{,}000$, $M = 10$, $O = 1000$ and $K = 10$. For the rewards resulting from the simulations, the 2500 observations are smoothed using a window of 10. For the policies Expl and Eps, the BAKF stepsize is used

with probability $p_d^D = (0.1, 0.8, 0.1)$, is already released for transportation (i.e., $r \in \mathcal{R} = \{0\}$ and $p_r^R = 1$), and has time-window length $k \in \mathcal{K} = \{0, 1, 2\}$ with probability $p_k^K = (0.2, 0.3, 0.5)$.

The costs are defined as follows. The long-haul, high capacity vehicle costs (per subset of destinations visited) are $C_{\mathcal{D}'} = (250, 350, 450, 900, 600, 700, 1000)$ for $\mathcal{D}' = (\{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\})$, respectively. These costs are for the entire long-haul vehicle, independent on the number of freight consolidated. Furthermore, we consider there are no costs for the long-haul vehicle if no freights are consolidated. The alternative, low capacity mode costs (per freight) are $B_d = (500, 1000, 700)$ for $d \in \mathcal{D}$. There is no discount factor, i.e. $\gamma = 1$.

We build three different sets of features based on a common "job" description used in transportation settings: *MustGo*, *MayGo*, and *Future* freights. MustGo freights are those released freights whose due-day is immediate. MayGo freights are those released freights whose due-day is not immediate. Future freights are those that have not yet been released. We use the MustGo, MayGo and Future adjectives in destinations as well, with an analogous meaning to those of freight. In Table 3.4 we show the three sets of features, which we name Value Function Approximation (VFA) 1, 2, and 3. All feature types in this table are related to the freights of a post-decision state. The symbol '*' denotes a VFA set containing a feature type. All feature types are numerical, and either indicate (i.e., 1 if yes, 0 if no), count (1,2,...), number (add), or multiply (i.e., product between two numbers) the different type of freights and destinations. Between parentheses we show the number of basis functions (i.e., independent variables) that a feature type has for the test instance. For example, there is one post-decision state variable per destination, per time-window length, thus all post-decision state variables are $3 * 3 = 9$. The constant feature equals one for all post-decision states, and the weights θ_a^n are all initialized with one.

Table 3.4: Various sets of features (basis functions of a post-decision state)

Feature type	VFA 1	VFA 2	VFA 3
All post-decision state variables (9)	*	*	*
All post-decision state variables squared (9)	*	–	–
Count of MustGo destinations (1)	*	*	*
Number of MustGo freights (1)	*	*	*
Product of MustGo destinations and MustGo freights (1)	*	–	–
Count of MayGo destinations (1)	*	*	*
Number of MayGo freights (1)	*	*	*
Product of MayGo destinations and MayGo freights (1)	*	–	–
Count of Future destinations (1)	*	*	*
Number of Future freights (1)	*	*	*
Product of Future destinations and Future freights (1)	*	–	–
Indicator MustGo freights per destination (3)	–	*	–
Indicator MayGo freights per destination (3)	–	*	–
Indicator Future freights per destination (3)	–	*	–
Number of all freights (1)	*	*	*
Constant (1)	*	*	*

References

1. R. Bellman, *Dynamic Programming*, 1st edn. (Princeton University Press, Princeton, NJ, 1957)
2. D.P.D. Farias, B.V. Roy, On constraint sampling in the linear programming approach to approximate dynamic programming. Math. Oper. Res. **29**(3), 462–478 (2004)
3. A.P. George, W.B. Powell, Adaptive stepsizes for recursive estimation with applications in approximate dynamic programming. Mach. Learn. **65**(1), 167–198 (2006)
4. A.P. George, W.B. Powell, S.R. Kulkarni, S. Mahadevan, Value function approximation using multiple aggregation for multiattribute resource management. J. Mach. Learn. Res. **9**, 2079–2111 (2008)
5. T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning*. Springer Series in Statistics (Springer, New York, NY, 2001)
6. P.J.H. Hulshof, M.R.K. Mes, R.J. Boucherie, E.W. Hans, Patient admission planning using approximate dynamic programming. Flex. Serv. Manuf. J. **28**(1), 30–61 (2016)
7. D.R. Jiang, T.V. Pham, W.B. Powell, D.F. Salas, W.R. Scott, A comparison of approximate dynamic programming techniques on benchmark energy storage problems: does anything work?, in *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, 2014, pp. 1–8
8. M.R.K. Mes, W.B. Powell, P.I. Frazier, Hierarchical knowledge gradient for sequential sampling. J. Mach. Learn. Res. **12**, 2931–2974 (2011)

9. A. Pérez Rivera, M.R.K. Mes, Dynamic multi-period freight consolidation, in *Computational Logistics*, ed. by F. Corman, S. Voß, R.R. Negenborn. Lecture Notes in Computer Science, vol. 9335 (Springer, Cham, 2015), pp. 370–385
10. W.B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley Series in Probability and Statistics (Wiley, London, 2011)
11. W.B. Powell, Perspectives of approximate dynamic programming. Ann. Oper. Res. **241**(1), 319–356 (2012)
12. W.B. Powell, Clearing the jungle of stochastic optimization, in *Informs Tutorials in Operations Research*, chap. 4 (INFORMS, Hanover, MD, 2014), pp. 109–137
13. W.B. Powell, I.O. Ryzhov, *Optimal Learning and Approximate Dynamic Programming* (Wiley, London, 2013), pp. 410–431
14. W.B. Powell, H.P. Simao, B. Bouzaiene-Ayari, Approximate dynamic programming in transportation and logistics: a unified framework. EURO J. Transp. Logist. **1**(3), 237–284 (2012)
15. I.O. Ryzhov, W.B. Powell, Approximate dynamic programming with correlated bayesian beliefs, in *Proceedings of the 48th Allerton Conference on Communication, Control and Computing* (2010)
16. R.S. Sutton, A.G. Barto, *Introduction to Reinforcement Learning*, 1st edn. (MIT Press, Cambridge, MA, 1998)
17. J.N. Tsitsiklis, B. Roy, Feature-based methods for large scale dynamic programming. Mach. Learn. **22**(1), 59–94 (1996)
18. W. van Heeswijk, M.R.K. Mes, M. Schutten, An approximate dynamic programming approach to urban freight distribution with batch arrivals, in *Computational Logistics*, ed. by F. Corman, S. Voß, R.R. Negenborn. Lecture Notes in Computer Science, vol. 9335 (Springer, Cham, 2015), pp. 61–75

Chapter 4
Server Optimization of Infinite Queueing Systems

András Mészáros and Miklós Telek

Abstract The problem of optimizing Markovian models with infinitely or finite but infeasible large state space is considered. In several practically interesting cases the state space of the model is finite and extremely large or infinite, and the transition and decision structures have some regular property which can be exploited for efficient analysis and optimization. Among the Markovian models with regular structure we discuss the analysis related to the birth death and the quasi birth death (QBD) structure.

4.1 Introduction

Queueing systems with discrete customers and infinite buffer form stochastic models with (countable) infinite state space. The problem of optimal control of such infinite queueing systems often occurs in practical applications. E.g., with the currently more and more widespread used of cloud computing resources the problem of optimal assignment of tasks or task fragments to service blocks is a very hot research topic.

One of the motivating examples of the current work is to find optimal server selection in a Markovian, work conserving (no server is idle when there is a waiting customer), multi server service unit when the servers might have temporal

A. Mészáros (✉)
MTA-BME Information Systems Research Group, Magyar Tudósok Körútja 2,
1117 Budapest, Hungary
e-mail: meszarosa@hit.bme.hu

M. Telek
Budapest University of Technology and Economics, Magyar Tudósok Körútja 2,
1117 Budapest, Hungary
e-mail: telek@hit.bme.hu

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_4

differences. In such a system with n servers the work conserving service policy defines the service process as long as there are at least n customers in the system, because the n oldest customers (assuming ordered service starts) have to be under service at the n servers. In contrast, when there are less then $n-1$ customers in the system and a new customer arrives the customer has to be directed to one of the idle servers. This choice of the idle server allows the optimization of the system behavior when the servers are at least temporarily different (for a graphical representation see Fig. 4.1).

The dominant property of this motivating example is that an infinite state Markov model needs to be controlled such that decisions are possible only in a finite set of states. We use Markov Decision Processes (MDPs) for optimal control of such systems and investigate the special properties of the MDPs with infinite states and finite set of states with possible decisions.

Markov Decision Processes (MDPs) are prevalent for analysing decision problems in queueing systems. The MDP methodology can be used to find the exact optimum in many cases, however, with increasing the size of the examined system its computation time may become prohibitively large. Furthermore, if the system contains an infinite buffer, the standard MDP solution algorithms are not applicable anymore. However, there are cases when these systems can still be analyzed using the tools developed for finite MDPs. There are some general properties that often hold for MDP solutions. Perhaps the most fundamental of them is the threshold form of the optimal policy. A policy is of threshold form, if the optimal decision on a state can be determined by comparing a certain parameter of the state to a fixed value (called threshold). For instance accepting requests to a queue may be optimal until the queue length reaches a certain value. See e.g. [6] or [3] for more examples.

Apart from exact optimal solutions, one can get a quasi-optimal solution by using certain approximation techniques. One possible approach is the truncation of the state space. This may happen based on the physical model (e.g. the size of the buffer is constrained) as in [9] and [5] for example. Alternatively one can use only mathematical considerations as discussed by Altman [1]. Another interesting approach is shown in [8], where a so-called deterministic simulative model is introduced. The essence of this model is that the original MDP is transformed in such a way that transitions of the new model all become deterministic.

Here we discuss another approach, the exact solution of MDP models with infinite or finite but large state spaces. We apply general results from Markov chain theory, e.g. the analysis of Markov chains measures associated with some subsets of states, which has been studied for a long time [2]. Based on the subset measures we introduce a Markov chain transformation with the replacement of one subset, which results in a smaller, thus more easily computable MDP model with the same optimal policy. For the application of this approach one needs to compute subset measures for subsets of infinitely many states if the original model is infinite, which is not possible in general, but there are cases when the regularity in the transition structure of the MDP can be exploited to compute the required subset measures.

The proposed methodology is used to compute the optimal control of some queueing systems. We study queueing systems with Poisson as well as with Markov

modulated arrivals and a shared infinite queue with multiple (identical or different) Markovian servers and investigate the following question: *If there are multiple idle servers and there is a request to be served, which server do we choose to serve this request to obtain optimal system operation?*

In the following we present the specifics of the aforementioned transformation method and its application for some concrete examples. The rest of the chapter is organized as follows. Section 4.4 summarizes the basics of MDPs and the elements of the Markov chain transformation method including the computation of subset measures in general and for some special cases with regular Markov chain structures like the birth death structure and the quasi birth death structure. A set of examples and their analysis based on the proposed Markov chain transformation method are presented in Sect. 4.5. Throughout this chapter we are going to build on some basic queueing knowledge, like queue, server, buffer, Poisson process, Little law, work conserving service, ….

4.2 Basic Definition and Notations

In this section we restrict the scope of the paper, introduce the applied notations for MDPs and refer to some classical results that will be used later. In the following we will only consider continuous time homogeneous MDPs without discount. Thus we will use the following definition for MDPs

Definition 4.1. Let us consider a process $X(t)$ on a continuous time Markov chain with state space S, a set of decisions $A = \{a_i\}$, a set of decision dependent generator matrices $Q = \{Q(a)|a \in A\}$ and a set of decision and state dependent cost rates $C = \{c^a(s)|a \in A, s \in S\}$. We say that the tuple (S, A, Q, C) is a continuous time Markov decision process.

In the following sometimes the C^a cost rate matrix will be used, which is a diagonal matrix constructed from the cost rates for decision a, such that

$$C^a{}_{i,j} = \begin{cases} c^a(i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

In this work we concentrate on optimizing for infinite horizon. Because there is no discount in the considered MDPs the goal function is the average cost rate of the process, i.e., the optimal strategy is

$$\pi^* = \arg\min_{\pi} \mathrm{E}_{\pi} \left[\lim_{k \to \infty} \frac{1}{T} \int_{t=0}^{T} c^{\pi(X(t))}(X(t)) dt \right], \tag{4.1}$$

which is known to be the same as

$$\pi^* = \arg\min_{\pi} \sum_{s \in S} \alpha^{\pi}(s) c^{\pi(s)}(s), \tag{4.2}$$

where $c^{\pi(s)}(s)$ is the cost rate in state s if the strategy is π and $\alpha^{\pi}(s)$ is the steady state probability of being in state s for policy π.

We mention here that the previous description stands for pure strategies (i.e. we always make the same decision in a state with 1 probability). As shown in [4], there always exists a pure strategy that gives the optimum for the average reward rate problem.

We also note that, even though we only consider continuous time MDP examples, the same results hold for the discrete time counterparts. The method to related the continuous and the discrete time processes is referred to as uniformization. The discrete time counterpart of a continuous time MDP can be obtained by $P = \frac{1}{\Delta(Q)}Q + I$, where P is the transition matrix of the discrete time MDP and $\Delta(Q)$ is the largest absolute value in matrix Q, that is $\Delta(Q) = \max_{i,j}(|Q_{i,j}|)$.

4.3 Motivating Examples

4.3.1 Optimization of a Queueing System with Two Different Servers

Let us consider an M/M/2 queueing system, i.e. a system with Poisson arrival process with parameter λ and two servers with exponential service times and parameters μ_1 and μ_2 respectively, see Fig. 4.1. We assume a shared infinite queue and investigate the following question: If both servers are idle and there is a request to be served, which server do we choose to serve this request to obtain optimal system operation? An intuitive measure of optimality is the average expected sojourn time (system time) $\mathrm{E}(T)$, which is the sum of the average expected waiting time and service time.

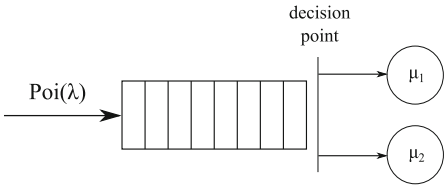


Fig. 4.1: M/M/2 queueing system with two different servers

We will utilize Little's law, which states that, $\mathrm{E}(n) = \bar{\lambda}\,\mathrm{E}(T)$, where $\mathrm{E}(n)$ is the expected value of average number of requests in the system and $\bar{\lambda}$ is the mean arrival intensity (in this case $\bar{\lambda} = \lambda$). Using this we will optimize $\mathrm{E}(n)$ as it is equivalent to the optimization of $\mathrm{E}(T)$ in the considered example because the decisions do not affect $\bar{\lambda}$.

We can write

$$\mathrm{E}(n) = \sum_{i=0}^{\infty} \alpha_i n(i), \tag{4.3}$$

where α_i is the steady state probability of state i and $n(i)$ is the number of requests in state i. By comparing this with the formula for average reward rate in (4.2), we can see that the problem can be formalized as an average reward rate optimization using $c^a(i) = n(i)$.

In the example we consider work conserving schemes only. This means that the service of any request has to start as soon as there is an idle server. Consequently there is only one decision in the system: when a new request arrives to the empty queue we have to decide whether server 1 or server 2 should serve this request.

The generator matrix of the MDP corresponding to this system is

$$Q^a = \left(\begin{array}{cccc|cc} -\lambda & p_a\lambda & (1-p_a)\lambda & 0 & \cdots & \\ \mu_1 & -\lambda-\mu_1 & 0 & \lambda & 0 & \cdots \\ \mu_2 & 0 & -\lambda-\mu_2 & \lambda & 0 & \cdots \\ 0 & \mu_2 & \mu_1 & -\lambda-\mu_1-\mu_2 & \lambda & 0 \ \cdots \\ \hline \vdots & 0 & 0 & \mu_1+\mu_2 & -\lambda-\mu_1-\mu_2 & \lambda \ \ddots \\ & \vdots & \ddots & \ddots & \ddots & \ddots \ \ddots \end{array}\right), \tag{4.4}$$

The single decision of choosing between server 1 and 2 happens in the first state. In Q^a this decision is represented by p_a, which is the probability of choosing server 1 upon arrival of a new request in the empty state, that is, the two possible decisions are always choosing the first server ($a = 1$) and always choosing the second server ($a = 2$) with $p_1 = 1$ and $p_2 = 0$. We recall here that there always exists a pure optimal strategy, therefore one of these decisions is optimal.

The cost of each state is the actual number of requests in the system; consequently,

$$c^a(i) = \begin{cases} 0, & \text{for } i = 1 \\ 1, & \text{for } i = 2 \\ i-2, & \text{otherwise} \end{cases} \tag{4.5}$$

for $a = 1, 2$. Note that the decisions do not affect the costs in this case, only the transitions.

4.3.2 Optimization of a Computational System with Power Saving Mode

In the second example we consider a system that executes simple computational tasks that can be decomposed to two steps, see Fig. 4.2. The steps take an exponentially distributed time with μ_1 and μ_2 parameter respectively. Tasks arrive according to a Poisson process of parameter λ. Usage of resources induces a certain cost per

time unit. Each waiting task requires the same amount of memory, generating cost with rate c_m, while the usage of the CPU generates cost with rate c_c. If the computer becomes idle it can either enter power saving mode, or remain in normal mode, which will be associated with c_i cost rate (power saving mode is assumed to have 0 cost rate). If a new task arrives while the computer is in power saving mode, the first part of the task takes an exponential time of μ_0 ($\mu_0 < \mu_1$) parameter. In other words, power saving mode costs less when the system is empty, but provides a slower service of the first request, which results in higher average CPU and memory usage costs. The operator has to decide if it is beneficial to use power saving mode.

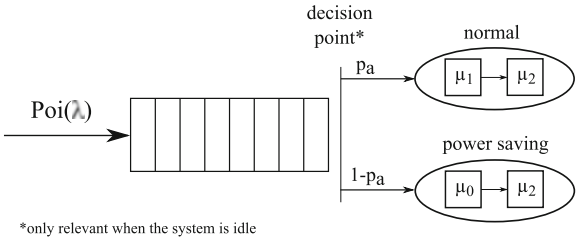


Fig. 4.2: Computational system with power saving mode

If state 1 corresponds to the empty system and p_a represents the decision of power saving such that $p_a = 1$ if power saving mode is used and $p_a = 0$ if not, then the generator of this process can be written as

$$Q^a = \left(\begin{array}{c|ccc|ccc|ccc|c}
-\lambda & p_a\lambda & (1-p_a)\lambda & 0 & 0 & 0 & 0 & & & & \\ \hline
0 & -\lambda-\mu_0 & 0 & \mu_0 & \lambda & 0 & 0 & & & & \\
0 & 0 & -\lambda-\mu_1 & \mu_1 & 0 & \lambda & 0 & & & & \\
\mu_2 & 0 & 0 & -\lambda-\mu_2 & 0 & 0 & \lambda & & & & \\ \hline
0 & 0 & 0 & 0 & -\lambda-\mu_0 & 0 & \mu_0 & \lambda & 0 & 0 & \\
0 & 0 & 0 & 0 & 0 & -\lambda-\mu_1 & \mu_1 & 0 & \lambda & 0 & \\
0 & \mu_2 & 0 & 0 & 0 & 0 & -\lambda-\mu_2 & 0 & 0 & \lambda & \\ \hline
 & & & & 0 & 0 & 0 & & & & \ddots \\
 & & & & 0 & 0 & 0 & & \ddots & & \\
 & & & & \mu_2 & 0 & 0 & & & & \\ \hline
 & & & & & & & \ddots & & &
\end{array}\right). \tag{4.6}$$

4.3.3 Structural Properties of These Motivating Examples

The main characteristics of the above described examples is associated with number of states of the queueing systems and the number of states where different decisions are possible. In both examples the overall state space is composed by infinitely many states, which inhibits the application of several standard MDP solution methods. On the other hand, the set of states in which decisions can be made (different actions can be chosen) is finite. These characteristic properties suggests the division of the set of states of such MDPs into two parts, the subset where decisions can be made and the complementer subset. Assuming that this structural properties are often present in MDP problems below we first introduce analysis results of Markov chains associated with state space division, and based on them we discuss a solution method of such MDPs.

4.4 Theoretical Background

In this part we briefly summarize the notations and the basic mathematical structures used for the decomposition based analysis of the considered MDP models.

4.4.1 Subset Measures in Markov Chains

The analysis of Markov chain properties associated with disjoint subsets of states has been considered for a very long time [2]. We summarize the related results in this subsection based on [7]. For a more detailed explanation of the presented results the reader is referred to that textbook. We borrow the terminology from reliability theory, where the operational states are commonly denoted as up states and the failure states as down states, and apply an S_U, S_D state partitioning such that $S_U \cup S_D = S$ and $S_U \cap S_D = \emptyset$. With appropriate numbering of states (states with low indexes are in S_U) the associated partitioning of the generator matrix is

$$Q = \begin{pmatrix} Q_U & Q_{UD} \\ Q_{DU} & Q_D \end{pmatrix}. \tag{4.7}$$

There are various interesting performance measures associated with the sets S_U and S_D. Let $\gamma_U = \min(t|X(t) \in S_U)$ be the time to reach a state in S_U. Starting from state $i \in S_D$ the joint distribution of the time to reach S_U and the state first visited in S_U is

$$\Theta_{ij}(t) = Pr(X(\gamma_U) = j, \gamma_U < t | X(0) = i) \tag{4.8}$$

The associated density function is $\theta_{ij}(t) = \frac{d}{dt}\Theta_{ij}(t)$ and the matrix function of size $|S_D| \times |S_U|$ composed by these elements satisfies

$$\theta(t) = \{\theta_{ij}(t)\} = e^{Q_D t} Q_{DU}.$$

Several interesting performance measures can be derived from this joint distribution. For example, the distribution of the state first visited in S_U is obtained as

$$\{Pr(X(\gamma_U) = j | X(0) = i)\} = \lim_{t\to\infty} \Theta(t) = \int_{t=0}^{\infty} \theta(t)dt = (-Q_D)^{-1} Q_{DU}, \quad (4.9)$$

where $i \in S_D$ and $j \in S_U$.

The inverse of matrix Q_D and Q_U always exist if the Markov chain is irreducible and positive recurrent, which we will assume in the following. The elements of matrix $(-Q_D)^{-1}$ have important stochastic meaning related to the time spent in the states of Q_D during a visit to S_D, that is for $i, j \in S_D$

$$E(\text{time spent in state } j \text{ in } (0,\gamma_U)|X(0) = i)) = E(\int_t I_{\{X(t)=j, \gamma_U > t | X(0)=i\}} dt)$$
$$= \int_t Pr(X(t) = j, \gamma_U > t | X(0) = i)dt = [\int_t e^{Q_D t} dt]_{ij} = [(-Q_D)^{-1}]_{ij}$$

where $(0,\gamma_U)$ is the time interval of the visit to S_D, $I_{\{\bullet\}}$ is the indicator of event $\bullet$ and $[M]_{ij}$ refers to the i,j element of matrix M. The time to reach S_U starting from state $i \in S_D$ is phase type distributed with the following density function

$$\sum_{j\in S_U} \theta_{i,j}(t) = e_i \theta(t) \mathbb{1} = e_i e^{Q_D t} Q_{DU} \mathbb{1}, \quad (4.10)$$

where e_i is the ith unit row vector, i.e. a vector with all its elements being zero except for the ith element which is one, and $\mathbb{1}$ is the column vector with all elements equal to one. To simplify the notations instead of scalar equations we often use appropriate vector expressions. For example (4.10) can be written as

$$\theta(t)\mathbb{1}_U = e^{Q_D t} Q_{DU} \mathbb{1}_U.$$

The size of vector $\mathbb{1}$ is determined by the context (the size of the matrix it is multiplied with), but occasionally we emphasize the dimension by a subscript. For example $\mathbb{1}_U$ refers to the vector of size $|S_U|$. One can obtain the $S_U \to S_D$ counterparts of these measures by interchanging the role of S_U and S_D in the above expressions.

Based on the joint distribution (4.8), for later use, we also present the conditional mean time spent in S_D supposing that the first state visited in S_U is j. For $i \in S_D$ and $j \in S_U$

$$E(\gamma_U | X(0) = i, X(\gamma_U) = j) = \frac{E(\gamma_U I_{\{X(\gamma_U)=j\}} | X(0) = i)}{Pr(X(\gamma_U) = j | X(0) = i)} = \quad (4.11)$$
$$= \frac{[\int_{t=0}^{\infty} t\theta(t)dt\mathbb{1}]_{ij}}{[\int_{t=0}^{\infty} \theta(t)dt\mathbb{1}]_{ij}} = \frac{[(-Q_D)^{-2} Q_{DU}]_{ij}}{[(-Q_D)^{-1} Q_{DU}]_{ij}}.$$

Let α be the stationary probability vector of the Markov chain with generator Q. Then α is the solution of the linear system $\alpha Q = 0$ with normalizing equation $\sum_{i \in S} \alpha_i = \alpha \mathbb{1} = 1$. Let α_U and α_D be the parts of vector α associated with subsets S_U and S_D respectively. Using (4.7) the partitioned form of the linear system is

$$\alpha_U Q_U + \alpha_D Q_{DU} = 0 \text{ and } \alpha_U Q_{UD} + \alpha_D Q_D = 0,$$

from which we obtain a linear system for α_U

$$\alpha_U (Q_U - Q_{UD} {Q_D}^{-1} Q_{DU}) = 0. \tag{4.12}$$

The Markov chain with state space S_U and generator $Q_U + Q_{UD}(-Q_D)^{-1} Q_{DU}$ is referred to as censored Markov chain. It is obtained from the original Markov chain by switching off the clock when the Markov chain visits S_D and switching on the clock when the Markov chain visits S_U.

The censored Markov chain defines the stationary probability of the states in S_U through (4.12) apart from a normalizing constant, because $\sum_{i \in S_u} \alpha_i = \alpha_U \mathbb{1}_U$ is not known based on (4.12). Intuitively, (4.12) defines the direction of vector α_U, but does not define its norm. To compute the norm $||\alpha_U|| = \alpha_U \mathbb{1}_U$ we calculate the time spent in S_U and S_D in consecutive visits. Let $T_U(n)$ ($T_U(n)$) be the time of the nth visit to S_U (S_D) and let us denote its limit by $T_U = \lim_{n \to \infty} T_U(n)$ ($T_D = \lim_{n \to \infty} T_D(n)$). The portion of time spent in S_U defines the norm of α_U by the following relation

$$\alpha_U \mathbb{1}_U = \frac{E(T_U)}{E(T_U) + E(T_D)} = \frac{1}{1 + \frac{E(T_D)}{E(T_U)}} .$$

$E(T_U)$ can be obtained as the inverse of the stationary rate form S_U to S_D, that is

$$E(T_U) = \frac{1}{\alpha_U Q_{UD} \mathbb{1}} ,$$

and $E(T_D)$ can be computed from the distribution in (4.10), where the ν_D initial distribution in S_D is characterized by the stationary distribution in S_U and a state transition from S_U to S_D, that is

$$E(T_D) = \nu_D \int_{t=0}^{\infty} t\theta(t) dt \mathbb{1} = \frac{\alpha_U Q_{UD}}{\alpha_U Q_{UD} \mathbb{1}} \int_{t=0}^{\infty} t\theta(t) dt \mathbb{1} = \tag{4.13}$$

$$= \frac{\alpha_U Q_{UD}}{\alpha_U Q_{UD} \mathbb{1}} (-Q_D)^{-2} Q_{DU} \mathbb{1} = \frac{\alpha_U Q_{UD}}{\alpha_U Q_{UD} \mathbb{1}} (-Q_D)^{-1} \mathbb{1} = \tag{4.14}$$

$$= E(T_U) \alpha_U Q_{UD} (-Q_D)^{-1} \mathbb{1},$$

where we used $(-Q_D)^{-1} Q_{DU} \mathbb{1} = \mathbb{1}$, which comes from the fact that the row sum of matrix Q is zero, that is $Q_{DU} \mathbb{1} + Q_D \mathbb{1} = 0$. Dividing the last expression by $E(T_U)$ gives

$$\alpha_U \mathbb{1}_U = \frac{E(T_U)}{E(T_U)+E(T_D)} = \frac{1}{1-\alpha_U Q_{UD} {Q_D}^{-1} \mathbb{1}} \,. \tag{4.15}$$

4.4.2 Markov Chain Transformation

There are practically interesting cases when the analysis of some performance measures is essentially related to only one subset of the states, say subset S_U. (As it is discussed below, in the context of MDPs we are going to consider cases when decisions can be made only in a subset of the states and the considered optimization problem is such that no decision is made in the rest of the sates.) In these cases it is possible to modify the Markov chain in the other subset, S_D, such that the important performance measures associated with S_U remain unchanged. For example, if we are interested only in α_U, the stationary distribution in S_U, it is possible to introduce a modified Markov chain with generator

$$\hat{Q} = \begin{pmatrix} Q_U & \hat{Q}_{UD} \\ \hat{Q}_{DU} & \hat{Q}_D \end{pmatrix}, \tag{4.16}$$

such that the stationary distribution $\hat{\alpha}$ is identical with the original stationary distribution α for the subset S_U that is $\hat{\alpha}_U = \alpha_U$.

The following example demonstrates this case.

Example 4.1. Let us consider the infinite birth-death Markov chain with birth rate λ, death rate μ and $S_U = \{0,1,\ldots,n-1\}$, $S_D = \{n,n+1,\ldots\}$. We introduce $\hat{S}_D = \{n\}$ with associated matrix blocks

$$\hat{Q}_{UD} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \lambda \end{pmatrix}, \;\; \hat{Q}_D = \begin{pmatrix} -\mu+\lambda \end{pmatrix}, \;\; \hat{Q}_{DU} = \begin{pmatrix} 0 \ldots 0 \; \mu-\lambda \end{pmatrix}.$$

The stationary distribution in S_U is identical for this modified Markov chain and the original one.

The Markov chain transformation in this example is rather intuitive because it retains the following essential properties

- The only possible transition from S_U to S_D ($\hat{S}_D$) is the transition from state $n-1$ to state n.
- The mean time spent in S_D, which is $\frac{1}{\mu-\lambda}$, is identical with the mean time spent $\hat{S}_D$.
- The only possible transition from S_D ($\hat{S}_D$) to S_U is the transition from state n to state $n-1$.

However, these simple properties do not have to hold in general. The following theorem provides a general rule for a Markov chain transformation which maintains the stationary distribution in a subset of states.

Theorem 4.1. *The stationary distribution of the Markov chain with generator Q and with generator $\hat{Q}$ are identical for S_U if the following conditions hold*

$$Q_{UD}(-Q_D)^{-1}Q_{DU} = \hat{Q}_{UD}(-\hat{Q}_D)^{-1}\hat{Q}_{DU} \tag{4.17}$$

and

$$Q_{UD}(-Q_D)^{-1}\mathbb{1} = \hat{Q}_{UD}(-\hat{Q}_D)^{-1}\mathbb{1}\,. \tag{4.18}$$

Proof. The linear system that characterizes the direction of α_U according to (4.12) is identical with the one characterizing the direction of $\hat{\alpha}_U$ based on $\hat{Q}$ due to (4.17). In order to ensure the identity of the α_U and $\hat{\alpha}_U$, we still need the sums of the stationary probabilities in S_U to be identical in the two systems, that is $\alpha_U\mathbb{1}_U = \hat{\alpha}_U\mathbb{1}_U$, which comes from (4.18) using (4.15).

In addition to the stationary distribution in a wide range of applications (including MDPs) it is important to maintain reward measures as well.

Theorem 4.2. *The stationary reward rate of a Markov reward model with generator Q and reward rate matrix C and with generator $\hat{Q}$ and reward rate matrix $\hat{C}$ are identical if* (4.17), (4.18), *$C_U = \hat{C}_U$ and the following condition holds*

$$Q_{UD}(-Q_D)^{-1}C_D\mathbb{1} = \hat{Q}_{UD}(-\hat{Q}_D)^{-1}\hat{C}_D\mathbb{1}. \tag{4.19}$$

Proof. The stationary reward rate in the modified Markov reward model is

$$\begin{aligned}\hat{\alpha}\hat{C}\mathbb{1} &= \hat{\alpha}_U\hat{C}_U\mathbb{1}_U + \hat{\alpha}_D\hat{C}_D\hat{\mathbb{1}}_D = \hat{\alpha}_U(\hat{C}_U\mathbb{1}_U + \hat{Q}_{UD}(-\hat{Q}_D)^{-1}\hat{C}_D\hat{\mathbb{1}}_D)\\ &= \alpha_U(C_U\mathbb{1}_U + Q_{UD}(-Q_D)^{-1}C_D\mathbb{1}_D) = \alpha C\mathbb{1}\end{aligned}$$

where we used $\hat{\alpha}_D = \hat{\alpha}_U\hat{Q}_{UD}(-\hat{Q}_D)^{-1}$ in the second equation and $\hat{\alpha}_U = \alpha_U$ (which comes from Theorem 4.1) in the third equation.

According to Theorems 4.1 and 4.2 one can replace a Markov chain with generator Q with a Markov chain with generator $\hat{Q}$ if the required performance measures are associated only with the stationary probabilities in S_U, and (4.17) and (4.18) hold. This replacement remains valid for reward measures as well if (4.19) holds additionally.

We note that (4.17) is about the identity of two matrices of size $|S_U| \times |S_U|$ and the rank of those matrixes is

$$r = \operatorname{rank}(Q_{UD}(-Q_D)^{-1}Q_{DU}) = \min(\operatorname{rank}(Q_{UD}), \operatorname{rank}(Q_{UD})). \tag{4.20}$$

Consequently the size of the transformed Markov chain should be at least $|S_U| + r$. For example, in Example 4.1 we have $r = 1$, because $\operatorname{rank}(\hat{Q}_{DU}) = \operatorname{rank}(\hat{Q}_{UD}) = 1$ and the transformed Markov chain has $n+1$ states.

4.4.3 Markov Decision Processes with a Set of Uncontrolled States

The above discussed state space division based analysis approaches can be efficiently used for the analysis of MDPs where decisions are possible only in a subset of states. More precisely, when there are states in the Markov chain where the Q^a_{ij} transition rates and the $c^a(i)$ associated cost are independent of the decision, that is $Q^a_{ij} = Q_{ij}$ and $c^a(i) = c(i)$, $\forall a \in A$. Unfortunately the efficient application of the space division depends on the properties of the considered problem. We consider some special cases below.

4.4.3.1 Decisions Only in Subset$_1$ Without an Effect on the Transitions to Subset$_2$

If the MDP is such that decisions are made only in subset$_1$ and it has no effect on the transitions to subset$_2$, then the generator matrix has the form

$$Q^a = \begin{pmatrix} Q^a_1 & Q_{12} \\ Q_{21} & Q_2 \end{pmatrix}.$$

In this case we can apply the association subset$_1$=S_U and subset$_2$=S_D and use the results of Theorems 4.1 and 4.2 in order to obtain a simple MDP problem with generator matrix

$$Q^a = \begin{pmatrix} Q^a_1 & \hat{Q}_{12} \\ \hat{Q}_{21} & \hat{Q}_2 \end{pmatrix}.$$

4.4.3.2 Decisions Only in Subset$_1$ with an Effect on the Transitions to Subset$_2$

If the MDP is such that decisions are made only in subset$_1$ and it has effect on the transitions to subset$_2$ then the generator matrix has the form

$$Q^a = \begin{pmatrix} Q^a_1 & Q^a_{12} \\ Q_{21} & Q_2 \end{pmatrix}.$$

In this case we can apply the association subset$_1$=S_U but we need to use the following decision dependent version of Theorem 4.1.

Theorem 4.3. *The stationary reward rate of the MDP with generator and reward matrix*

$$Q^a = \begin{pmatrix} Q^a_U & Q^a_{UD} \\ Q_{DU} & Q_D \end{pmatrix}, \quad C^a = \begin{pmatrix} C^a_U & 0 \\ 0 & C_D \end{pmatrix},$$

and the MDP with generator and reward matrix

$$\hat{Q}^a = \begin{pmatrix} Q^a_U & \hat{Q}^a{}_{UD} \\ \hat{Q}_{DU} & \hat{Q}_U \end{pmatrix}, \quad \hat{C}^a = \begin{pmatrix} C^a_U & 0 \\ 0 & \hat{C}_D \end{pmatrix},$$

are identical for any policy if the following conditions hold

$$Q^a_{UD}{Q_D}^{-1}Q_{DU} = \hat{Q}^a{}_{UD}{\hat{Q}_D}^{-1}\hat{Q}_{DU}, \tag{4.21}$$

$$Q^a_{UD}{Q_D}^{-1}\mathbb{1} = \hat{Q}^a{}_{UD}{\hat{Q}_D}^{-1}\mathbb{1}, \tag{4.22}$$

and

$$Q^a_{UD}{Q_D}^{-1}C_D\mathbb{1} = \hat{Q}^a{}_{UD}{\hat{Q}_D}^{-1}\hat{C}_D\mathbb{1}. \tag{4.23}$$

Proof. The proof of Theorem 4.3 directly follows from the proofs of Theorems 4.1 and 4.2.

4.4.3.3 Decisions Only in Subset$_1$ with Limited Boundary to the Other Set

If the MDP is such that decisions are made only in subset$_1$ but the transitions from subset$_1$ towards the rest of the states can reach only a part of the complementer subset without decision, denoted as subset$_2$, and the remaining part of the subset without decision, denoted as subset$_3$, cannot be reached from subset$_1$, then the generator matrix has the form

$$Q^a = \begin{pmatrix} Q^a_1 & Q^a_{12} & 0 \\ Q_{21} & Q_2 & Q_{23} \\ Q_{31} & Q_{32} & Q_3 \end{pmatrix}.$$

In this case we can apply the association subset$_1 \cup$ subset$_1$=S_U and subset$_3$=S_D and with these set definitions the results of Theorems 4.1 and 4.2 are directly applicable again.

4.4.4 Infinite Markov Chains with Regular Structure

Thanks to Theorems 4.1–4.3 Markov chain transformations where the original and the transformed problem have different sizes can be applied in the analysis of MDPs with a set of uncontrolled states. These transformations can be efficiently used when the original problem has a finite or even infinite state space. In this work we focus on the application of Markov chain transformation methods with infinite state space. In case of general infinite state MDPs with completely irregular structure the application of Theorems 4.1–4.3 is rather difficult, but in the majority of the practically interesting cases infinite state MDPs have some regular structure. We consider two of the simplest structures below.

4.4.4.1 Birth Death Process

An MDP has a birth-death structure when (with appropriate numbering of states) state transitions are possible only to neighboring states. A birth-death structure can contain level dependent and level independent rates. Example 4.1 discusses the case of level independent rates. Here we focus on the level dependent case. Let the arrival and departure rates at state $k < n$ be $\lambda_k(a)$ and μ_k and at state $k \geq n$ be λ_k and μ_k respectively. Furthermore let $S_U = \{0, 1, \ldots, n-1\}$ and $S_D = \{n, n+1, \ldots\}$. Similar to Example 4.1 we can transform the MDP such that $\hat{S}_D = \{n\}$ with associated matrix blocks

$$\hat{Q}^a{}_{UD} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \lambda_{n-1}(a) \end{pmatrix}, \quad \hat{Q}_D = \begin{pmatrix} -\hat{\mu} \end{pmatrix}, \quad \hat{Q}_{DU} = \begin{pmatrix} 0 \ldots 0 \ \hat{\mu} \end{pmatrix}.$$

The rate from $\hat{S}_D$ to S_U, $\hat{\mu}$, can be computed form the recursive relation on the mean time spent in the set $S_k = \{k, k+1, \ldots\}$, denoted by T_k, that is

$$T_k = \frac{1}{\lambda_k + \mu_k} + \frac{\lambda_k}{\lambda_k + \mu_k} T_{k+1}$$

where $\hat{\mu} = \frac{1}{T_n}$. If λ_k and μ_k are independent of k then this relation results in $\hat{\mu} = \mu - \lambda$ as in Example 4.1. If λ_k and μ_k are state dependent then the recursive relation needs to be solved based on the specific form of state dependence. Finally the unknown reward rate $\hat{c}$ can be computed based on (4.19).

4.5 Solution and Numerical Analysis of the Motivating Examples

In this section we provide some specific examples for the usage of the transformation techniques presented in the previous section.

4.5.1 Solution to the Queue with Two Different Servers

As marked in (4.4) we select the first four states as S_U and the rest as S_D.

Notice that the upper part of this system is a birth death process, thus we can use the results from Example 4.1 to get

$$\hat{Q}^a = \left(\begin{array}{cccc|c} -\lambda & p_a\lambda & (1-p_a)\lambda & 0 & \\ \mu_1 & -\lambda-\mu_1 & 0 & \lambda & 0 \\ \mu_2 & 0 & -\lambda-\mu_2 & \lambda & 0 \\ 0 & \mu_2 & \mu_1 & -\lambda-\mu_1-\mu_2 & \lambda \\ \hline 0 & 0 & 0 & \mu_1+\mu_2-\lambda & -\mu_1-\mu_2+\lambda \end{array}\right). \tag{4.24}$$

We chose $S_U = \{1,2,3,4\}$ and $S_D = \{5\}$, as it is indicated in the transition rate matrix. We can apply the previously presented cost transformation in (4.42) by noticing that this system is a special QBD where $G = [1]$, i.e., it is a 1×1 matrix with its only element being 1, from which Z and consequently $C_{i\to j}$ can be calculated. By substituting into (4.38) and using notation $\mu = \mu_1 + \mu_2$ we obtain

$$[\hat{C}^a]_{5,5} = \frac{\sum_{i=0}^{\infty} \frac{1}{\mu}(\frac{\lambda}{\mu})^i (i+3)}{\sum_{i=0}^{\infty} \frac{1}{\mu}(\frac{\lambda}{\mu})^i}.$$

We can use $\sum_{i=0}^{\infty} x^i = \frac{1}{1-x}$ and $\sum_{i=0}^{\infty} i x^i = \frac{x}{(1-x)^2}$ to simplify the expression and get the modified cost function

$$[\hat{C}]_{i,i} = \begin{cases} 0, & \text{for } i = 1 \\ 1, & \text{for } i = 2,3 \\ 2, & \text{for } i = 4 \\ 3 + \frac{\lambda}{\mu-\lambda}, & \text{for } i = 5 \end{cases} \tag{4.25}$$

The MDP described by $\hat{Q}^a$ and $\hat{C}$ can be solved using standard solution algorithms. Let us consider a specific example with $\lambda = 10$, $\mu_1 = 1$, $\mu_2 = 100$. Using these values we get $\mathrm{E}(n) = 0.15$ for $a = 1$ ($p_a = 0$) and $\mathrm{E}(n) = 0.19$ for $a = 2$ ($p_a = 1$). Unsurprisingly the optimal decision is choosing the faster server whenever it is possible. In this example the optimal strategy is trivial. It can be shown analytically that choosing the faster server is always optimal. For more complex systems; however, giving an analytical solution may be impossible.

4.5.2 Solution to the Power-Saving Model

Starting from state 5 the generator is a QBD with block independent transition rates. Thus we will transform the MDP while keeping the first five states unchanged, that is, we choose $S_U = \{1,\ldots,5\}$.

While this problem is more complicated than the previous one, we can exploit an important structural characteristic to transform the system to finite states without the usage of the matrix analytic methodology. We will create the same additional states as with the previously proposed transformation method in Sect. 4.7.2, but use elementary arguments to obtain the $\omega_{i,j}$ and $[G]_{i,j}$ parameters in the transition rates. Let use notation $\tau_{k,j} = E(\gamma_U | X(0^+) = k, X(\gamma_U) = j)$, $k \in S_D$, $j \in S_U$. From (4.36)

it is clear that $\omega_{i,j} = \sum_{k \in S_D} \frac{Pr(X(0^+)=k|X(0^-)=i}{Pr(X(0^+ \in S_D)} \tau_{k,j}$. Thus, if we can calculate $\tau_{k,j}$, $\omega_{i,j}$ can be calculated as well.

Note that Q^a has a QBD structure with group independent blocks starting from group 2. Let us denote the ith state of group n by (n,i). Because of the block independent QBD structure of the generator we can write

$$\tau_{(n,i)\to(n-1,j)} = \tau_{(n+1,i)\to(n,j)}, \forall n > 1,\ i,j = 1,2,3, \tag{4.26}$$

that is, the time to reach state j of group $n-1$ from state i of group n does not depend on the actual value of n. Furthermore note that the states of group n can only be reached from higher groups through state $(n,1)$ for $n \geq 1$. Consequently $\tau_{(n,i)\to(1,2)}$ can be expressed as

$$\tau_{(n,i)\to(1,2)} = \tau_{(2,1)\to(1,2)} + \tau_{(3,1)\to(2,1)} + \cdots + \tau_{(n,1)\to(n-1,1)} + \tau_{(n,i)\to(n,1)} \tag{4.27}$$

We can write recursive relations similar to the one for birth death processes. For example

$$\tau_{(2,1)\to(1,2)} = \frac{\lambda}{\lambda+\mu_0}\left(\frac{1}{\lambda+\mu_0} + \tau_{(3,1)\to(1,2)}\right) + \frac{\mu_0}{\lambda+\mu_0}\left(\tau_{(2,3)\to(1,2)} + \frac{1}{\lambda+\mu_0}\right). \tag{4.28}$$

Here the first term is the expected time it takes to reach state 2 from state 5 if the first event is the arrival of a new request weighted by the probability $\frac{\lambda}{\lambda+\mu_0}$ of such an event. The second term corresponds to the other possibility, i.e., the current request is served before a new request arrives. The probability of this event is $\frac{\mu_0}{\lambda+\mu_0}$. In this case the expected time to reach state 2 is $\frac{1}{\lambda+\mu_0} + \tau_{(2,3)\to(1,2)}$. We can derive expressions for $\tau_{(2,2)\to(1,2)}$ and $\tau_{(2,3)\to(1,2)}$ using the same approach. Thus we get

$$\tau_{(2,2)\to(1,2)} = \frac{\lambda}{\lambda+\mu_1}\left(\frac{1}{\lambda+\mu_1} + \tau_{(2,2)\to(1,2)}\right) + \frac{\mu_1}{\lambda+\mu_1}\left(\frac{1}{\lambda+\mu_1} + \tau_{(2,3)\to(1,2)}\right) \tag{4.29}$$

$$\tau_{(2,3)\to(1,2)} = \frac{\lambda}{\lambda+\mu_2}\left(\frac{1}{\lambda+\mu_2} + \tau_{(3,3)\to(1,2)}\right) + \frac{\mu_2}{\lambda+\mu_2}\frac{1}{\lambda+\mu_2}. \tag{4.30}$$

Furthermore, from (4.27) we have $\tau_{(3,1)\to(1,2)} = \tau_{(3,1)\to(2,1)} + \tau_{(2,1)\to(1,2)}$, $\tau_{(3,2)\to(1,2)} = \tau_{(3,2)\to(2,1)} + \tau_{(2,1)\to(1,2)}$, $\tau_{(3,3)\to(1,2)} = \tau_{(3,3)\to(2,1)} + \tau_{(2,1)\to(1,2)}$, additionally we have $\tau_{(3,2)\to(2,1)} = \tau_{(2,2)\to(1,2)}$, $\tau_{(3,3)\to(2,1)} = \tau_{(2,3)\to(1,2)}$. Using these the attained linear equation system can be easily solved for $\tau_{(2,1)\to(1,2)}, \tau_{(2,2)\to(1,2)}, \tau_{(2,3)\to(1,2)}$, however it results in rather complicated expressions, therefore we do not present the actual solutions. From S_D we always reach S_U in state 2, thus we only need to introduce states $\hat{s}_{2\to2}$, $\hat{s}_{3\to2}$, and $\hat{s}_{4\to2}$ to create a transformed version of the MDP, and for these we only need the previously given τ parameters. Furthermore, from the definition of G it is clear that

$$[G]_{i,j} = \begin{cases} 1, \text{ if } j = 2 \\ 0, \text{ otherwise} \end{cases} \tag{4.31}$$

Thus we have the necessary τ (and consequently ω) and G values to calculate the elements of $\hat{Q}_{UD}$, $\hat{Q}_D$ and $\hat{Q}_{DU}$ using formulas (4.35) and (4.37).

The original cost function of this system is

$$[C^a]_{i,i} = \begin{cases} pc_i, & \text{for } i = 1 \\ c_n + \lfloor \frac{i+2}{3} \rfloor c_m, & \text{for } i \geq 2. \end{cases} \tag{4.32}$$

To calculate the modified costs we can use the same (4.45) formula as in the case of the M/M/2 system, utilizing (4.31) to calculate Z.

Let us take an example where the request arrival rate and service rates are $\lambda = 5$, $\mu_0 = 2$, $\mu_1 = 10$, $\mu_2 = 20$, the cost rate of not entering power saving mode is $c_i = 20$ and the cost rate of memory consumption and CPU usage are $c_m = 2$, $c_n = 10$ respectively. In this case we get

$$\hat{Q}_{UD} = \begin{pmatrix} 5 & 0 & 0 \\ 5 & 0 & 0 \\ 5 & 0 & 0 \end{pmatrix}, \quad \hat{Q}_{DU} = \begin{pmatrix} 1.67 & 0 & 0 \\ 0.45 & 0 & 0 \\ 5 & 0 & 0 \end{pmatrix}, \quad \hat{C}_D = \begin{pmatrix} 28.6 & 0 & 0 \\ 32.5 & 0 & 0 \\ 28 & 0 & 0 \end{pmatrix}$$

and the average cost rate is approximately 38.5 if we use power saving mode and 17.5 if we do not, which means that power saving mode should not be used.

The example did not require the usage of numerical methods for calculating G. The main reason for this is that, when the request was served, the system could go to only one state. Consequently the structure of the G matrix was very special and its values could be derived using elementary arguments. For the same reason the calculation of $\hat{Q}_{UD}$ and $\hat{Q}_{DU}$ could also be done using elementary tools. In the following examples the structure of the generator becomes even more complex, thus the usage of the previously presented will be necessary.

4.6 Further Examples

In the remaining examples we will examine queueing systems with a Markov background process. The point process with Markov background process is referred to as Markov Arrival Process (MAP). The series of inter event times of a MAP form a dependent series of the random variables (in general). We use this series as the consecutive service times of a server, which is some times referred to as Markov Service Process, or MAP service times. The states of the Markov background process are often referred to as phases.

Definition 4.2. Markov Arrival Process (MAP) is a point process modulated by a background Markov chain. The transition rates which modify the state of the background Markov chain but are not associated with an event of the point process are collected into matrix S_0 and the transition rates which might or might not modify the state of the background Markov chain and are associated with an event of the point process are collected into matrix S_1. The diagonal elements of S_0 are defined

such that $Q = S_0 + S_1$ is the generator of the background Markov chain (with zero row sums).

MAPs form a quite general framework for modeling point processes with different correlation structure and marginal distributions while making a simple description and analysis of the overall stochastic model possible.

4.6.1 Optimization of a Queuing System with Two Markov Modulated Servers

First let us consider a two server system very much like in the first example, with the only difference being that the servers are identical and they perform service according to a Markov Arrival Process. To avoid confusion we will call the state of a service MAP "phase", and retain the term "state" for the states of the MDP. (We recall again that the events of the MAP are the service events in this case.) We presume that the internal state of a server (the phase of the MAP) may only change if that server is not idle. Otherwise our assumptions are the same as before: we assume a Poisson arrival process with parameter λ and a shared infinite queue and investigate the following question: If both servers are idle and there is a request to be served, which server do we choose to serve this request to obtain optimal system operation? We choose the average expected sojourn time as the measure of optimality but work with the expected value of the average number of requests in the system which are proportional according to Little's law for the same reason as in the M/M/2 example. Consequently the cost of each state is the number of requests for that state ($C(i) = n(i)$) just like in the M/M/2 example. Also in this case we restrict our inspection to work conserving schemes.

4.6.2 Structural Properties of the Example with Markov Modulated Servers

The state transition structure of the MDP describing the behavior of the queuing system with two Markov modulated servers is different from the birth-death structure of the previous examples, because apart of the number of customers in the system the system state has to contain information about the "phase" of the Markov modulated servers. With a proper lexicographical numbering of states the set of states with identical number of customers are continuously indexed (an are commonly referred to as "level"). Due to the fact that a transition can change the number of customers in the system at most by one nonzero transition rates are possible only between neighboring levels. Introducing matrix blocks that contain the state transitions between

levels we obtain a similar birth death structure as in (4.24) on the level of matrix blocks. This transition matrix structure is referred to Quasi birth death structure and is studied in the next section.

4.7 Infinite MDPs with Quasi Birth Death Structure

4.7.1 Quasi Birth Death Process

Another regular structure of infinite MDPs with practical interest is the quasi birth death (QBD) structure [7] (all results of this subsection are available in [7]). The QBD structure is a generalization of the birth death structure, where the states are divided into groups of finite sizes and transitions are possible only inside a group and between neighboring groups. If the states are numbered according to increasing group identifiers then the transition matrix has the form

$\mathbf{L_0}$	$\mathbf{F_0}$			
$\mathbf{B_1}$	$\mathbf{L_1}$	$\mathbf{F_1}$		
	$\mathbf{B_2}$	$\mathbf{L_2}$	$\mathbf{F_2}$	
		$\mathbf{B_3}$	$\mathbf{L_3}$	$\mathbf{F_3}$
			$\ddots$	$\ddots$

,

where $\mathbf{L_k}$ contains the transitions inside group k, $\mathbf{F_k}$ contains the transitions from group k to group $k+1$, $\mathbf{B_k}$ contains the transitions from group k to group $k-1$, and the idle blocks indicate blocks with zero elements. The size of the groups might be different, but $\mathbf{L_k}$ is an invertible square matrix if the Markov chain is irreducible and positive recurrent.

We introduce a partitioning based on the groups of the QBD. Let sets $S_1, S_2, \ldots$ be defined such that S_n contains the states of group n. Then matrix $G_n(t)$ describes the joint distribution of time to reach S_{n-1} and the state visited first in S_{n-1} starting from a state in S_n. A similar joint distribution is described by matrix $\Theta(t)$ in (4.8), but here matrix $G_n(t)$ corresponds to the group based partitioning of the QBD.

$$[G_n(t)]_{i,j} = Pr(X(\gamma_{n-1}) = j, \gamma_{n-1} < t | X(0) = i),\ i \in S_n,\ j \in S_{n-1}, \tag{4.33}$$

where, like before, $\gamma_n = min(t | X(t) \in S_n)$.

The transform domain expressions for $G_n(t)$ is

$$sG_n(s) = B_n + L_n G_n(s) + F_n G_{n+1}(s) G_n(s)$$

from which the distribution of the state visited first in group $n-1$ is the solution of the recursive equation

$$0 = B_n + L_n G_n + F_n G_{n+1} G_n$$

and the measure related with the mean time to reach group $n-1$, $G'_n = \lim_{s\to 0} \frac{d}{ds} G_n(s)$, can be obtained from

$$G_n = L_n G'_n + F_n G'_{n+1} G_n + F_n G_{n+1} G'_n.$$

There are rather few practically interesting cases when the solution of this recursive equation is available for group dependent transition rates. In practical applications the case of group independent transition rates is much more common.

If the transition rates are block independent, that is, $B_k = B$, $L_k = L$, $F_k = F$ ($\forall k \geq n$), then the matrix expressions simplify to

$$0 = B + LG + FG^2 \tag{4.34}$$

and

$$G = LG' + FG'G + FGG'.$$

The first one is a quadratic matrix equation whose minimal non-negative solution can be computed by efficient numerical procedures. When G is known, the second equation is a Sylvester equation for G'.

One of the fundamental statements of group independent QBD theory is that the steady state probability of states has a matrix geometric distribution, i.e.

$$\alpha_{n+1} = \alpha_n R,$$

where α_n is a vector containing the steady state probabilities of states in S_n. Matrix R can be calculated from G as

$$R = F(-L - FG)^{-1}.$$

In the next section we use G, R and other associated matrices to transform MDPs whose uncontrolled set has a (group independent) QBD structure.

4.7.2 Solving MDPs with QBD Structure

In this subsection we present a specific method for the transformation of MDPs with a set of uncontrolled states using the partitioning of Sect. 4.4.3.1.

When the uncontrolled QBD blocks are of size n, the rank of matrix $Q_{UD}{Q_D}^{-1}Q_{DU}$ in (4.20) is at most n. In this section we present a Markov chain transformation method which maintains the steady state reward rate of the MDP according to Theorems 4.1 and 4.2 The new Markov chain is such that during a given visit to $\hat{S}_D$ only a single state is visited before the transition back to S_U. The key idea of the transformation is to assign a state in the transformed MDP to each possible transition from S_U to S_U through a visit in S_D. Matrix $Q_{UD}(-Q_D)^{-1}Q_{DU}$ is

composed of a single (potentially) non-zero block of size $n \times n$ associated with the $S_U \to S_D \to S_U$ transition from the last block of S_U to the same block, since transitions are possible only to the neighboring blocks. This non-zero matrix block is composed of n^2 elements. We introduce a modified MDP such that $\hat{S}_D$ is composed of n^2 elements. The associated $\hat{Q}_{UD}$, $\hat{Q}_D$, $\hat{Q}_{DU}$, are defined as follows. Each of $\hat{Q}_{UD}$, $\hat{Q}_D$ and $\hat{Q}_{DU}$ contain (at most) one non-zero elements per row. It means that transition $i \in S_U \to S_D \to j \in S_U$ is described with a $i \in S_U \to \hat{S}_D \to j \in S_U$ transition where the only state visited in $\hat{S}_D$ is associated with the described $i \in S_U \to S_D \to j \in S_U$ transition and is denoted by $s_{i \to j}$. See Fig. 4.3.

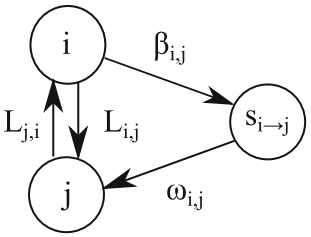


Fig. 4.3: Transitions in the transformed Markov chain: $i, j \in S_U$, $s_{i \to j} \in \hat{S}_D$

There are (at most) n^2 such state transitions and associated states. If transition $i \in S_U \to S_D \to j \in S_U$ is impossible for a given pair of states in the last block of S_U then impossible state transitions and associated $s_{i \to j}$ states can be eliminated from $\hat{S}_D$, which results in less than n^2 states in $\hat{S}_D$.

The transition rate from i to $s_{i \to j}$ is the i, j element of the matrix block in $Q_{UD}(-Q_D)^{-1}Q_{DU}$ associated with the last block of S_U, that is

$$\beta_{ij} = [Q_{UD}(-Q_D)^{-1}Q_{DU}]_{ij} \,. \tag{4.35}$$

The transition rate from $s_{i \to j}$ to j is computed based on the conditional mean time spent in S_D supposed that the process moves to S_D from state i and the first state visited in S_U is j. When the initial state in S_D is known this quantity is provided in (4.11). In our case we need to consider the distribution of the initial state in S_D as well. For $i, j \in S_U$

$$\begin{aligned} &E(\gamma_U | X(0^-) = i, X(0^+) \in S_D, X(\gamma_U) = j) = \\ &= \frac{E(\gamma_U I_{\{X(\gamma_U)=j\}} | X(0^-) = i, X(0^+) \in S_D)}{Pr(X(\gamma_U) = j | X(0^-) = i, X(0^+) \in S_D)} = \frac{[Q_{UD}(-Q_D)^{-2}Q_{DU}]_{ij}}{[Q_{UD}(-Q_D)^{-1}Q_{DU}]_{ij}} \,. \end{aligned} \tag{4.36}$$

The transition rate from $s_{i \to j}$ to j is the inverse of the conditional mean time in (4.36), that is

$$\omega_{ij} = \frac{[Q_{UD}(-Q_D)^{-1}Q_{DU}]_{ij}}{[Q_{UD}(-Q_D)^{-2}Q_{DU}]_{ij}} \,. \tag{4.37}$$

With this definition matrix $\hat{Q}_D$ is a diagonal matrix (with negative diagonal elements) and matrix $(-\hat{Q}_D)^{-1}\hat{Q}_{DU}$ is a kind of mapping matrix with only one nonzero element per row whose value is 1. The identity of $Q_{UD}(-Q_D)^{-1}Q_{DU}$ and $\hat{Q}_{UD}(-\hat{Q}_D)^{-1}\hat{Q}_{DU}$, which is required for Theorem 4.1 to hold, comes from the fact that the only nonzero element of $\hat{Q}_{UD}$ in the row associated with state i is equal with the appropriate element of $Q_{UD}(-Q_D)^{-1}Q_{DU}$ and the multiplication with matrix $(-\hat{Q}_D)^{-1}\hat{Q}_{DU}$ maps this element to the appropriate position.

The identity of $Q_{UD}(-Q_D)^{-1}\mathbb{1}$ and $\hat{Q}_{UD}(-\hat{Q}_D)^{-1}\mathbb{1}$, can be obtained as follows. Matrix $(-\hat{Q}_D)^{-1}$ is a diagonal matrix whose element associated with $s_{i\to j}$ is the expression on the right hand size of (4.36). The only non-zero matrix element of $\hat{Q}_{UD}$ associated with that state is $[Q_{UD}(-Q_D)^{-1}Q_{DU}]_{ij}$. The product of the two is $[Q_{UD}(-Q_D)^{-2}Q_{DU}]_{ij}$. When we sum up these quantities for all states in $\hat{S}_D$ we obtain $Q_{UD}(-Q_D)^{-2}Q_{DU}\mathbb{1} = Q_{UD}(-Q_D)^{-1}\mathbb{1}$.

The reward rate of state $s_{i\to j}$ is defined as

$$C_{i\to j} = \frac{[Q_{UD}(-Q_D)^{-1}C_D(-Q_D)^{-1}Q_{DU}]_{ij}}{[Q_{UD}(-Q_D)^{-2}Q_{DU}]_{ij}} \,. \tag{4.38}$$

We still need to show that the reward rates in $\hat{S}_D$ are defined such that they fulfill the conditions of Theorem 4.2. Since matrix $(-\hat{Q}_D)^{-1}$ is diagonal with diagonal elements given in (4.36) the product $(-\hat{Q}_D)^{-1}\hat{C}_D$ is also diagonal with diagonal elements $\frac{[Q_{UD}(-Q_D)^{-1}C_D(-Q_D)^{-1}Q_{DU}]_{ij}}{[Q_{UD}(-Q_D)^{-1}Q_{DU}]_{ij}}$. Multiplying this diagonal element with the i to $s_{i\to j}$ transition of matrix $\hat{Q}_{UD}$ we have $[Q_{UD}(-Q_D)^{-1}C_D(-Q_D)^{-1}Q_{DU}]_{ij}$. Summing up this quantity for the destination state j we have

$$Q_{UD}(-Q_D)^{-1}C_D(-Q_D)^{-1}Q_{DU}\mathbb{1} = Q_{UD}(-Q_D)^{-1}C_D\mathbb{1} \,.$$

We note that there are more than one reward rate definition which fulfills Theorem 4.2. The one in (4.38) is such that the above described $(Q,C) \to (\hat{Q},\hat{C})$ Markov reward model transformation does not modify the Markov reward model (apart of potential renumbering of states) whose original structure (defined by matrix Q) complies with the structure of matrix $\hat{Q}$ depicted in Fig. 4.3.

4.7.2.1 QBD Measures Associated Infinite Sets

Already Theorems 4.1 and 4.2 indicate that the Markov chain transformation approach is applicable only if we can compute the measures on the left hand size of (4.17)–(4.19). If S_D is composed by a finite number of states it is a trivial computational task with $\mathcal{O}(|S_D|^3)$ complexity. If S_D is composed by an infinite number of states it is a more difficult problem which has a nice solution only in a limited number of cases. One those cases is Markov chain with group independent QBD structure. In that case the Q_{UD} Q_D and Q_{DU} matrices have the following structure.

$$Q_{UD} = \begin{bmatrix} & & & \cdots \\ & & & \cdots \\ \mathbf{F} & & & \cdots \end{bmatrix}, \quad Q_D = \begin{bmatrix} \mathbf{L} & \mathbf{F} & & \\ \mathbf{B} & \mathbf{L} & \mathbf{F} & \\ & \mathbf{B} & \mathbf{L} & \ddots \\ & & \ddots & \ddots \end{bmatrix}, \quad Q_{DU} = \begin{bmatrix} & & \mathbf{B} \\ & & \\ & & \\ \vdots & \vdots & \vdots \end{bmatrix}.$$

Due to the block structure of matrix Q_D its inverse is a full matrix. When, to compute $Q_{UD}(-Q_D)^{-1}Q_{DU}$, we multiply this full matrix with Q_{UD} from the left and with Q_{DU} from the right only the upper left block of $(-Q_D)^{-1}$ plays role in the result and that block is computable based on the process restricted to the first group S_D [7] as $(-L-FG)^{-1}$. The essential main value of this expression is that a block of an infinite matrix inverse can be computed by a finite matrix inverse. Consequently the only non-zero block of matrix $Q_{UD}(-Q_D)^{-1}Q_{DU}$, its lower left block, equals to FZB, where matrix Z is defined as $Z = (-L-FG)^{-1}$. That is

$$[Q_{UD}(-Q_D)^{-1}Q_{DU}]_{ij} = [FZB]_{ij}\,, \tag{4.39}$$

where the left hand side of the equation refers to the i,j element of the non-zero block. We note that $R = FZ$ and $G = ZB$, which can be used to simplify the notations. Unfortunately, the computation of $Q_{UD}(-Q_D)^{-2}Q_{DU}$ requires the evaluation of further blocks of the infinite matrix inverse $(-Q_D)^{-1}$, because all blocks of the upper row and the left column of $(-Q_D)^{-1}$ contribute to the upper left block of $(-Q_D)^{-2}$. The kth block of the upper row of $(-Q_D)^{-1}$ is $Z(FZ)^{k-1} = ZR^{k-1}$, and the kth block of the left column of $(-Q_D)^{-1}$ is $(ZB)^{k-1}Z = G^{k-1}Z$. Using these relations, we have

$$[Q_{UD}(-Q_D)^{-2}Q_{DU}]_{ij} = [\sum_{k=1}^{\infty} R^k G^k]_{ij}\,, \tag{4.40}$$

where the infinite sum, $S = \sum_{k=1}^{\infty} R^k G^k$, can be computed by the following simple iterative procedure in Table 4.1. For positive recurrent Markov chains the infinite summation converges to a finite limit, because the spectral radius of R is less than 1 and the spectral radius of G is 1.

The block structure of the reward rate matrix is

$$C_D = \begin{bmatrix} C_{D1} & & \\ & C_{D2} & \\ & & \ddots \end{bmatrix},$$

where matrices C_{Dk} denote the reward rate matrix associated with the kth group of S_D. Utilizing the knowledge on the upper row of $(-Q_D)^{-1}$ the overall reward measure associated with a visit to S_D can be computed as

$X = I$;
$S = 0$;
Repeat
$X = RXG$;
$S = S + X$;
Until $\|X\| < \varepsilon$;

Table 4.1: Procedure 1

$X = I$; $Y = I$;
$S = 0$; $k = 1$;
Repeat
$X = RX$; $Y = YG$;
$S = S + XC_{Dk}Y$; $k++$;
Until $\|XC_{Dk}Y\| < \varepsilon$;

Table 4.2: Procedure 2

$$Q_{UD}(-Q_D)^{-1}C_D\mathbb{1} = \sum_{k=1}^{\infty} R^k C_{Dk}\mathbb{1} \ . \tag{4.41}$$

The evaluation of this infinite sum depends on the properties of the reward rate matrix. The infinite summation converges to a finite limit if the Markov chain is positive recurrent and the C_{Dk} series increases sub-exponentially. In practice, the most common case is when C_{Dk} is proportional to k which results in a finite limit for positive recurrent Markov chains.

If the reward rate is group dependent in S_D, then numerical iterations are required to compute the infinite summation. If the reward rate is group independent in S_D, that is $C_{Dk} = \bar{C}_D$ for $\forall k \geq 1$, then

$$Q_{UD}(-Q_D)^{-1}C_D\mathbb{1} = \sum_{k=1}^{\infty} (FZ)^k \bar{C}_D\mathbb{1} = FZ(I - FZ)^{-1}\bar{C}_D\mathbb{1} \ ,$$

which expression, on the right hand side, contains operations with computable finite matrices only. Assuming group dependent reward rates and utilizing again the upper row and the left column of $(-Q_D)^{-1}$, we have

$$[Q_{UD}(-Q_D)^{-1}C_D(-Q_D)^{-1}Q_{DU}]_{ij} = [\sum_{k=1}^{\infty} R^k C_{Dk} G^k]_{ij} \ , \tag{4.42}$$

where the infinite sum can be computed by the numerical procedure in Table 4.2. For positive recurrent Markov chains this infinite summation has the same convergence behavior as the one in (4.41).

Finally, we summarize the QBD specific measures of the Markov chain transformation for later use

$$\beta_{ij} = [FZB]_{ij} \ , \tag{4.43}$$

$$\omega_{ij} = \frac{[FZB]_{ij}}{[\sum_{k=1}^{\infty} R^k G^k]_{ij}} \ , \tag{4.44}$$

$$C_{i\to j} = \frac{[\sum_{k=1}^{\infty} R^k C_{Dk} G^k]_{ij}}{[\sum_{k=1}^{\infty} R^k G^k]_{ij}} \ . \tag{4.45}$$

4.8 Solution and Numerical Analysis of MDPs with QBD Structure

4.8.1 Solution of the Example with Markov Modulated Servers

Let us denote by S_0, S_1 the corresponding matrices of the MAP and use the standard $\otimes$ and $\oplus$ notation for the Kronecker product and Kronecker sum operators respectively. Furthermore let us denote by I_x the identity matrix of size x. Then the generator matrix of the describing MDP is

$$Q = \left(\begin{array}{cc|cccc} L_0 & F_0 & 0 & \cdots & & \\ B_1 & L_1 & F_1 & 0 & \cdots & \\ \hline 0 & B_2 & L & F & 0 & \\ \vdots & 0 & B & L & F & \ddots \\ & \vdots & \ddots & \ddots & \ddots & \ddots \end{array}\right). \tag{4.46}$$

The blocks of Q are

$$L_0 = -\lambda I_4, \quad F_0 = \lambda \left(P\, I_4 - P\right), \quad B_1 = \begin{pmatrix} I_2 \otimes S_1 \\ S_1 \otimes I_2 \end{pmatrix},$$
$$L_1 = \begin{pmatrix} I_2 \otimes S_0 - \lambda I_4 & 0 \\ 0 & S_0 \otimes I_2 - \lambda I_4 \end{pmatrix}, \quad F_1 = \lambda \begin{pmatrix} I_4 \\ I_4 \end{pmatrix},$$
$$B_2 = \left(S_1 \otimes I_2 \; I_2 \otimes S_1\right), \quad L = S_0 \oplus S_0 - \lambda I_4, \quad F = \lambda I_4,$$
$$B = S_1 \otimes I_2 + I_2 \otimes S_1,$$

where

$$P = \begin{pmatrix} z & 0 & 0 & 0 \\ 0 & p & 0 & 0 \\ 0 & 0 & 1-p & 0 \\ 0 & 0 & 0 & z \end{pmatrix}, \tag{4.47}$$

where z is an arbitrary value in $[0,1]$ The cost function is

$$C^a{}_{i,i} = \begin{cases} 0, & \text{for } 1 \geq i \leq 4 \\ 1, & \text{for } 5 \geq i \leq 12 \\ \lfloor \frac{i}{4} \rfloor c_m, & \text{otherwise .} \end{cases} \tag{4.48}$$

In this queueing system there is one simple question to be answered: If both servers are idle, one of them is in phase 1 and the other one is in phase 2, which server has to process the next arriving customer to have a minimal average system time? In the generator this decision is represented by p in matrix P. If $p = 1$ we choose the server in phase 1, if $p = 0$ we choose the server 2.

Let us take a specific example, where $\lambda = 10$ and

$$S_0 = \begin{pmatrix} -0.1 & 0.05 \\ 0 & -100 \end{pmatrix}, \quad \begin{pmatrix} 0.05 & 0 \\ 5 & 95 \end{pmatrix}. \tag{4.49}$$

and let $U = \{1,2,3,4\}$ as indicated by the partitioning in (4.46).

Based on intuition and the results of the M/M/2 system the optimal strategy is to choose the server which can serve the customer faster. This means that we compare the mean service time starting from phase 1 and phase 2, i.e., $t_1 = \underline{e_1}^T(-S_0)^{-1}\mathbb{1}$ and $t_2 = \underline{e_2}^T(-S_0)^{-1}\mathbb{1}$, and if the first expression is smaller, we choose the server in phase 1 ($p = 1$), otherwise the one in phase 2 ($p = 0$), in this case $t_1 = x$, $t_2 = x$, thus $p = 1$ should be optimal. If we solve the MDP, however, we find that $\mathrm{E}(n) \approx 0.11$ if $p = 0$ and $\mathrm{E} \approx 0.098$ if $p = 1$; i.e., it is better to choose the server which serves the customer slower. This counter-intuitive result can be interpreted the following way. If we use the faster server for the first customer, the probability of finishing the service before a new arrival is high, as the mean service time of the faster state is smaller than the mean inter-arrival time of a new customer. Upon service there is a chance that the server moves to the slower state, leaving the system with two servers in the phase with higher service time. In this state there is a higher chance that more than two consecutive customers arrive before the first customer can be served, which leads to a higher average system time. In other words, assigning the customer to the faster server leads to a more deteriorated state after service completion, while assigning the customer with the server in the slower phase, there is a chance that the server will move to the faster state upon service, thus the state of the system improves. One can think of this effect as the repair of the server at the cost of a slower service. Extensive numerical investigations suggest that choosing the server with higher service time is optimal for any M/MAP(2)/2 system regardless of the other characteristics of the service MAP and the intensity of arrivals.

4.8.2 Markov Modulated Server with Three Background States

In the previous example a simple—although counterintuitive—rule could be made for the optimal decision. For even slightly more complicated systems this becomes increasingly difficult. Let us take the same system as before just change the service MAP from a MAP(2) to a MAP(3):

$$S_0 = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -2.3 & 0 \\ 0 & 0 & -100 \end{pmatrix}, \quad S_1 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 2.3 \\ 100 & 0 & 0 \end{pmatrix}.$$

Solution of this system is done the same way as before. Let us set $\lambda = 1.2$. In this case the optimal strategy is to always prioritize the server in phase 1 and choose the server in phase 2 over the one in phase 3. This is in accordance with the results of the M/MAP(2)/2 case, i.e., we choose the slowest available server. For $\lambda = 1.5$, however, it is better to choose the server in phase 3, if the other server is in phase 2. This example demonstrates, that, even for very simple cases, the optimal strategy

cannot be determined based on intuition or simple examination of the system. In these cases the numerical solution of the problem is required.

4.9 Conclusion

We have considered the problem of numerical analysis of MDPs with very large and infinite state spaces, where decisions can be made only in a finite subset of states. To handle such MDP models we introduced a general framework for model transformations of MDP such that the modified model has the same optimal policy as the original one. The applicability of this framework depends on the computability of some performance measures associated with a subset of states of the MDP model. We presented the computation of those subset measures in case of two special Markov chain structures the birth death and the quasi birth death structure. We applied the proposed methodology for a set of application examples where the optimal control of queueing systems with infinite buffer is of interest.

Acknowledgements This work is partially supported by the OTKA K101150 projects.

References

1. E. Altman, *Constrained Markov Decision Processes*, vol. 7 (CRC, Boca Raton, FL, 1999)
2. J.A. Buzacott, Markov approach to finding failure times of repairable systems. IEEE Trans. Reliab. **R-19**(4), 128–134 (1970)
3. D. Efrosinin, Controlled queueing systems with heterogeneous servers. Ph.D. thesis, University of Trier, 2004
4. J. Filar, K. Vrieze, *Competitive Markov Decision Processes* (Springer, Berlin, 1997)
5. K. Jagannathan, S. Mannor, I. Menache, E. Modiano, A state action frequency approach to throughput maximization over uncertain wireless channels. Internet Math. **9**(2–3), 136–160 (2013)
6. Y. Kocaga, A. Ward, Admission control for a multi-server queue with abandonment. Queueing Syst. **65**(3), 275–323 (2010)
7. G. Latouche, V. Ramaswami, *Introduction to Matrix Analytic Methods in Stochastic Modeling*, vol. 5 (SIAM, Philadelphia, 1999)
8. A.Y. Ng, M. Jordan, Pegasus: a policy search method for large mdps and pomdps, in *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence* (Morgan Kaufmann, Los Altos, CA, 2000), pp. 406–415
9. J. Slegers, I. Mitrani, N. Thomas, Optimal dynamic server allocation in systems with on/off sources, in *Formal Methods and Stochastic Models for Performance Evaluation* (Springer, Berlin, 2007), pp. 186–199

Chapter 5
Structures of Optimal Policies in MDPs with Unbounded Jumps: The State of Our Art

H. Blok and F.M. Spieksma

Abstract The derivation of structural properties of countable state Markov decision processes (MDPs) is generally based on sample path methods or value iteration arguments. In the latter case, the method is to inductively prove the structural properties of interest for the n-horizon value function. A limit argument then should allow to deduce the structural properties for the infinite-horizon value function.
In the case of discrete time MDPs with the objective to minimise the total expected α-discounted cost, this procedure is justified under mild conditions. When the objective is to minimise the long run average expected cost, value iteration does not necessarily converge. Allowing time to be continuous does not generate any further complications when the jump rates are bounded as a function of state, due to applicability of uniformisation. However, when the jump rates are *unbounded* as a function of state, uniformisation is only applicable after a suitable perturbation of the jump rates that does not destroy the desired structural properties. Thus, also a second limit argument is required.
The importance of unbounded rate countable state MDPs has increased lately, due to applications modelling customer or patient impatience and abandonment. The theory validating the required limit arguments however does not seem to be complete, and results are scattered over the literature.
In this chapter our objective has been to provide a systematic way to tackle this problem under relatively mild conditions, and to provide the necessary theory validating the presented approach. The base model is a parametrised Markov process (MP):

H. Blok (✉)
Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
e-mail: h.blok@tue.nl

F.M. Spieksma
Leiden University, Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
e-mail: spieksma@math.leidenuniv.nl

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_5

both perturbed MPs and MDPs are special cases of a parametrised MP. The advantage is that the parameter can simultaneously model a policy and a perturbation.

5.1 Introduction

The question how to rigorously prove structural results for continuous time Markov decision problems (MDPs) with a countable state space and unbounded jump rates (as a function of state) seems to be an assiduous task. As a typical example one may consider the competing queues model with queue dependent cost rates per customer and per unit time, where the objective is to determine the server allocation that minimises the total expected discounted cost or expected average cost per unit time. Both discounted and average cost are known to be minimised by the $c\mu$-rule, which prescribes to allocate the server to the queue that yields the largest cost reduction per unit time. A possible method to tackle this problem is to apply value iteration (VI) to the uniformised discrete time MDP and show that optimality of the $c\mu$-rule propagates through the VI step.

If customer abandonment is allowed, the resulting MDP is a continuous time MDP with unbounded jumps, since customers may renege at a rate that is proportional to the number of customers present in each of the queues. In order to apply VI, one needs to time-discretise the MDP. One way to associate a discrete time MDP with this problem is by constructing the decision process embedded on the jumps of the continuous time MDP. However, it is not clear whether structural properties propagate through the VI step (cf. Sect. 5.3.4). Another solution is to perturb or truncate the continuous time MDP, so that it becomes uniformisable and then apply VI. A suitable truncation or perturbation needs to be invariant with respect to structural properties of interest of the investigated MDP.

The first question is whether there exist generic truncation methods that possess such an invariance property. Clearly, this can never be systematically proved, since it depends on the properties that one wishes to prove. However, one might be able to formulate recommendations as to what kind of perturbation methods perform well, with regard to such an invariance requirement. The paper [22] studies two competing queues with abandonments, and a problem-specific truncation is used. Later [8] has introduced a truncation method, called smoothed rate truncation (SRT) that so far seems to work well for problems where a simple bounded rate truncation (as in Sect. 5.3.4) does not work. In addition, it can be used for numerical applications in bounded rate optimisation problems (cf. Sect. 5.2.2). The SRT method has been used in a companion paper [9] for identifying conditions under which a simple index policy is optimal in the competing queues problem with abandonments.

Consecutively, suppose that an application of the truncation method yields truncated MDPs with optimal policies and a value function that have the properties of interest. For instance, the above mentioned application of SRT to the competing queues example with abandoning customers yields optimality of an index policy for each truncated MDP. However, these properties still have to be shown to ap-

ply to the original non-truncated problem. Thus, convergence results are required that yield continuity of the optimal policy and value function in the truncation parameter, in order to deduce the desired results for the non-truncated MDP from the truncated ones.

A second question therefore is as to what kind of easily verifiable conditions on the input parameters of the perturbed MDP guarantees convergence of the value function and optimal policy to the corresponding ones of the original unperturbed MDP. In [22], the authors had to prove a separate convergence result apart from devising a suitable truncation and prove that VI propagates the properties of interest. Apparently, theory based on a set of generic conditions that can incorporate convergence within the optimisation framework was lacking. This lack is precisely what has hampered the analysis of the server farm model in [1], where the authors have restricted their analysis to showing threshold optimality of a bounded rate perturbed variant of the original model. Apparently no appropriate tools for deducing threshold optimality of the original unbounded problem from the results for the perturbed one were available to them.

A third major problem occurs in the context of the average cost criterion. In particular, VI is not always guaranteed to converge. This is true under weak assumptions in discounted cost problems, however, in average cost problems there are only limited convergence results (cf. Sect. 5.2.3). One of these requires strong drift conditions, that do not allow transience under any stationary deterministic policy. However, often this is not a convenient requirement. One may get around this difficulty by a vanishing discount approach, which analyses the expected average cost as a limit of expected α-discounted costs as the discount factor tends to 0 (or 1, depending on how the discount factor is modelled).

For a model like the competing queues model with abandonments, a multistep procedure to obtain structural results for the average cost problem then would be as follows. First, consider the α-discounted cost truncated problem. Structural results for the α-discounted cost non-truncated problem follow, by taking the limit for the truncation parameter to vanish. Finally, taking the limit of the discount factor to 0 hopefully yields the final structural results for the original continuous time average cost problem.

For some of these steps theoretical validation has been provided for in the literature, but not for all, and not always under conditions that are easily checked. The main focus of this chapter is to fill some gaps in the described procedure, whilst requiring conditions that are formulated in terms of the input parameters. Based on the obtained results, we aim to provide a systematic and feasible approach for attacking the validation of structural properties, in the spirit of the multistep procedure sketched above. We hope that this multistep procedure will also be beneficial to other researchers as a roadmap for tackling the problem of deriving structural results for problems modelled as MDPs.

We do not address the methods of propagating structures of optimal policies and value function through the VI step. Such methods belong to the domain of 'event based dynamic programming', and they have been discussed thoroughly in [33], with extensions to SRT and other rate truncations in [11]. Furthermore, we

do not include an elaborate evaluation of closely related results from the literature. Some detailed comments has been included in this chapter, whenever we thought this relevant. We also would like to emphasize that the derivations in this chapter are not new: we mainly adapted earlier original proofs to fill the gaps that hampered application of the scheme we have presented in this chapter for deriving structural properties. We have always tried to refer to the appropriate reference containing the proof ideas that we have adapted.

Another omission in this work is the study of perturbed MDPs with the average cost criterion. However, the conditions required for achieving the desired continuity results as a function of a perturbation parameter are quite strong. Therefore a more recommendable approach would be the one we have developed in this chapter, using the vanishing discount approach. As a last remark: we generally restrict to the class of stationary policies, and not history-dependent ones. Especially the results quoted for discrete time MDPs apply to the larger policy class. In continuous time MDPs allowing history-dependent policies causes extra technical complications that we do not want to address in this work.

A short overview of the chapter content is provided next. In Sect. 5.2 we discuss discrete time, countable state MDPs, with compact action sets. First, the α-discount optimality criterion is discussed, cf. Sect. 5.2.1. This will be the base case model, to which the MDP problems might have to be reduced in order to investigate its structural properties. We therefore describe it quite elaborately. In addition, we have put it into a framework that incorporates truncations or perturbations. We call this a parametrised Markov process. Interestingly enough, 'standard' but quite weak drift conditions introduced for α-discounted cost MDPs in discrete time, allowed this extension to parametrised Markov processes, with no extra effort and restriction. It incorporates the finite state space case, elaborated on in the seminal book [20].

In Sect. 5.2.2 we provide a discussion of SRT, as a method for numerical investigation of structural properties of a countable state MDP. The conditions that we use are a weak drift condition on the parametrised process, plus reasonable continuity conditions. This has been based on the work in [34, 54] for MDPs.

In Sect. 5.2.3 we study the expected average cost criterion, whilst restricting to non-negative cost, i.e. negative dynamic programming. This restriction allows transience, and the analysis follows [15], in the form presented by Sennott [44]. Basically, the conditions imposed require the existence of one 'well-behaved' policy, and a variant of inf-compact costs. The latter ensures that optimal policies have a guaranteed drift towards a finite set of low cost states. The contribution of these works is that they validate the vanishing discount approach, thus allowing to analyse the discrete time average cost problem via the discrete time α-discounted cost problem.

Then we turn to studying continuous time MDPs in Sect. 5.3. First the α-discounted cost problem is considered. The drift conditions on parametrised discrete time Markov processes have a natural extension to continuous time. The results listed are based on [13], but the literature contains quite some work in the same spirit within the framework of MDPs with more general state spaces, cf. e.g. [27, 38], and references therein. A closely related perturbation approach has been studied in [37]. Since perturbations are incorporated in the parametrised framework, the ap-

proach allows to study bounded jump perturbations. Indeed, optimal policies and value functions are continuous as a function of the perturbation parameter. In this way, [13] obtains threshold optimality of the original unbounded α-discounted cost variant of the server farm model studied in [1].

Finally, for the expected average cost criterion, we use the natural generalisation of the discrete time conditions. Although closely related to analyses in [27, 38] and references therein, as far as we know this form has not appeared yet in the literature. The vanishing discount approach is validated in the same way as was done for the discrete time MDP. This reduces the problem of studying structural properties for average cost MDPs satisfying the proposed conditions, to analysing a continuous time α-discounted cost MDP, for which the solution method has already been described. As a consequence, also average cost threshold optimality for the afore mentioned server farm model from [1] follows from α-discount optimality of a threshold policy, cf. [13].

Dispersed through the chapter are roadmaps for the validation of structural properties. These are summarised in Sect. 5.3.4.

Notice that instead of the usual nomenclature 'Markov chain' for a Markov process in discrete time, we will consistently use 'Markov process', whether it be a process in discrete or continuous time. We use S as the notation for a countable state space, and $\mathbf{Z}_+ = \{0,1,\ldots\}$. The discount factor $\alpha \in (0,1)$ in discrete time models will mean that cost is discounted at rate $1-\alpha$, contrary to the general use. This allowed a more direct analogy with continuous time models.

5.2 Discrete Time Model

In order to illustrate the formal setup of a parametrised Markov process, we will first introduce a simple example motivating the use of parametrised processes instead of restricting to MDPs.

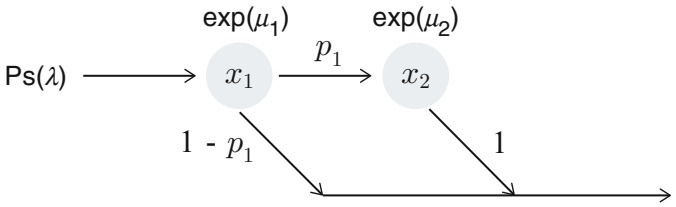


Fig. 5.1: Inspection/reparation centre

Example 1 (Server Allocation) *Consider the following server allocation problem, see Fig. 5.1. Products arrive at an inspection/repair centre according to a Poisson process with rate λ. Inspection takes an exponentially distributed amount of time*

with parameter μ_1. After finishing inspection in unit 1, with probability p_1 a repair at unit 2 is required, which takes an $\exp(\mu_2)$ distributed amount of time, $p_1 \in (0,1)$. With probability $1-p_1$ the product does not require any additional repair and it leaves the center after finishing inspection.

There is only one repair man, the server, who has to be allocated to one of the two units. We assume that idling is not allowed (no coffee breaks while there is work to do). Allocation is based on the number of products i_1 in the inspection phase unit 1, and the number of products i_2 in the repair phase unit 2. Thus the state space is $S = \mathbf{Z}_+^2$. The goal is to determine an allocation policy that minimises the overall number of products waiting for inspection and/or repair. The specific optimisation criteria that we will study, will be discussed below.

This problem can then be modelled as an MDP, for which optimal state-dependent allocation decisions may be computed through various methods.

However, for numerical computation of the optimal cost and allocation decisions, the state space needs to be truncated. A minimal requirement for a suitable truncation is convergence of the optimal cost and allocation decision of the truncated process to the ones of the original non-truncated process that we wish to analyse. Convergence properties are not difficult to obtain; mild conditions that ensure the existence of optimal policies and the convergence of suitable algorithms, also yield continuity properties of associated truncated MDPs. Thus it makes sense to incorporate control (which is essentially a state-dependent parameter) and truncation within one framework, which we will call a parametrised Markov process.

Note that truncations might affect the structural properties of the optimal allocation as a function of the state, see Fig. 5.2 in the continuation Example 8 of the analysis of the tandem queue.

We will next set up the framework of parametrised Markov processes in discrete time. To this end, let Φ be a parameter space. With each parameter ϕ associate $S \times S$ stochastic matrix P^ϕ, and a cost vector $c(\phi) : S \to \Re$. We denote the corresponding elements by $p^\phi(j\,|\,i)$, $x, y \in S$, and $c^\phi(i)$, $i \in S$. Let $f : S \to \Re$, and denote the value at $j \in S$ by $f(j)$. Then, $P^\phi f$ is the function with value

$$P^\phi f(i) = \sum_j p^\phi(j\,|\,i) f(j)$$

at point $i \in S$, provided the integral is well-defined. We further denote by $P_{i\cdot}^\phi$ the i-th row of P^ϕ $\phi \in \Phi$, $i \in S$.

To transition matrix P^ϕ one can associate a Markov process on the path space $\Omega = S^\infty$. Given a distribution ν on S, the Kolmogorov consistency theorem (see e.g. [10]) provides the existence of a probability measure P_ν^ϕ on Ω, such that the canonical process $\{X_n\}_n$ on Ω, defined by

$$X_n(\omega) = \omega_n, \quad \omega = (\omega_1, \ldots, \omega_n, \ldots)$$

is a Markov process with transition matrix P^{ϕ}, and probability distribution P_{ν}^{ϕ}. The corresponding expectation operator is denoted by E_{ν}^{ϕ}. We further denote $P^{\phi,(n)}$ for the n-th iterate of P^{ϕ}, where $P^{\phi,(0)} = \mathbf{I}$ equals the $S \times S$ identity matrix.

We assume the following basic assumption.

Assumption 1 *The following conditions hold:*

i) $\phi \mapsto p^{\phi}(j\,|\,i)$ *continuous on* Φ *for each* $i, j \in S$*;*
ii) $\phi \mapsto c^{\phi}(i)$ *is continuous on* Φ *for each* $i \in S$*.*
iii) the parameter space Φ *is locally compact;*

To characterise MDPs in this set up, we use the following concept.

Definition 5.1. We say that $\{P^{\phi}, c^{\phi}\}_{\phi\in\Phi}$ has the product property with respect to Φ, if for any $\phi^1, \phi^2 \in \Phi$, $i \in S$, there exists $\phi \in \Phi$ with

$$P_{j\cdot}^{\phi} = \begin{cases} P_{j\cdot}^{\phi^1}, & j = i \\ P_{j\cdot}^{\phi^2}, & j \neq i \end{cases} \qquad c^{\phi}(j) = \begin{cases} c^{\phi^1}(j), & j = i \\ c^{\phi^2}(j), & j \neq i; \end{cases} \tag{5.1}$$

For notational convenience we will simply say that Φ has the product property. In case the dependence on ϕ is expressed in the probability or expectation operators, we write $c(X_n)$ instead $c^{\phi}(X_n)$.

Thus, under the product property it is possible to 'exchange' the i-th rows of transition matrix P^{ϕ^2} and cost vector c^{ϕ^2} for the i-th rows of P^{ϕ^1}, and c^{ϕ^1} respectively, $\phi^1, \phi^2 \in \Phi$, without 'leaving' the parameter set Φ.

Remark 5.1. An MDP is usually set up by specifying an action set $\mathcal{A}(i)$ *in state i, and the associated transition probabilities from x as well as the immediate cost incurred in i, per action from* $\mathcal{A}(i)$*. Then the set of deterministic policies can be identified with* $\mathcal{A} = \prod_{i\in S}\mathcal{A}(i)$*. Putting* $\Phi := \mathcal{A}$ *yields a parametrised Markov process with a parameter set* Φ *with the product property. This is easily verified.*

Additionally, let $\Pi(i)$ *be the collection of all probability distributions on* $\mathcal{A}(i)$ *for each state* $i \in S$ *(details are beyond the scope of this paper). The collection* $\Pi = \prod_i \Pi(i)$ *of all stationary randomised policies can be viewed as a parameter set having the product property as well. We will not consider this explicitly, but all discussed results cover this case.*

If Φ *has the product property, then the parametrised Markov process is an MDP. This follows directly by putting* $\mathcal{A}(i) = \Phi$*,* $i \in S$*. This point of view is not new: in [29] MDPs have been set up as a collection of stochastic matrices with associated cost function with the product property.*

Example 2 (Continuation of Example 1 Server Allocation) *As discussed in the above, allocation decisions are based on the number of products in units 1 and 2. Therefore, the process state* $X_n = i = (i_1, i_2)$*, at time n, corresponds to* i_1 *products being present at time n in unit 1 and* i_2 *in unit 2,* $i = (i_1, i_2) \in S$*. The action spaces are*

$$\mathcal{A}(i) = \begin{cases} \{1,2\}, \text{ if } i_1, i_2 > 0; \\ \{1\}, \quad \text{if } i_2 = 0; \\ \{2\}, \quad \text{if } i_1 = 0; \\ \{0\} \quad \text{if } i_1 = i_2 = 0. \end{cases}$$

Put $\mathcal{A} = \prod_{i \in S} \mathcal{A}(i)$. Then, $\delta \in \mathcal{A}$ is a state-dependent server allocation rule, where $\delta(i) \in \{1,2\}$ is the allocation of the server to unit $\delta(i)$, when the system state is i.

For this model, we consider truncations that consist of 'deleting' the transitions leading from states inside rectangles of the form $R(N_1,N_2) = \{(i_1,i_2) \in \mathbf{Z}_+ \mid i_1 \leq N_1, i_2 \leq N_2\}$ out of the same rectangle, $N_1,N_2 = 1,2,\ldots$. Put

$$\Phi = \{\phi \,|\, \phi = (N_1,N_2,\delta) \,|\, N_1,N_2 \in \mathbf{Z}_+ \cup \{\infty\}, \delta \in \mathcal{A}\}.$$

The transition probabilities associated with each parameter are obtained by uniformisation, where we scale $\lambda + \mu_1 + \mu_2 = 1$. Simultaneous events cannot take place within a time-slot after this procedure, since the discrete time process described is an embedded jump process where the jump rates have been uniformised over states and potential decisions. For $\phi = (N_1,N_2,\delta)$ this leads to:

$$p^{\phi}(j \,|\, i) = \begin{cases} \lambda & j = (i_1+1, i_2), \quad i_1 \neq N_1 \\ p_1\mu_1 & j = (i_1-1, i_2+1), \; i_1 > 0, i_2 \neq N_2, \delta(i) = 1, \\ (1-p_1)\mu_1 & j = (i_1-1, i_2), \quad i_1 > 0, \delta(i) = 1 \\ \mu_2 & j = (i_1, i_2-1), \quad i_2 > 0, \delta(i) = 2 \\ 1 - \sum_{j' \neq i} p^{\phi}(j' \,|\, i) & j = i. \end{cases}$$

In order that the truncated process lives on all of S, we have not interfered with the transition probabilities outside the rectangle $R(N_1,N_2)$. Note that $R(N_1,N_2)$ is a finite and closed set, which is reached under any allocation rule with probability 1. We would like to point out that the allocation action sets are not affected by the truncation parameter in this model. In a similar manner, other truncations can be incorporated in the parametrised setting as well.

The cost vector considered here is

$$c^{\phi}(i) = i_1 + i_2, \quad \phi \in \Phi,$$

as the quality of the process is judged by the overall number of products in the two units.

Fixing the truncation parameter (N_1,N_2) leads to the parameter set

$$\Phi_{N_1,N_2} = \{\phi \in \Phi \,|\, \phi = (N_1,N_2,\delta), \delta \in \mathcal{A}\} = \{(N_1,N_2)\} \times \prod_{i \in S} \mathcal{A}(i).$$

Φ_{N_1,N_2} corresponds to the collection of deterministic server allocations for the (N_1,N_2)-truncated process, and it has the product property. Thus, it is an MDP. Parameter set $\Phi_{\infty,\infty}$ corresponds to the set of stationary, deterministic policies for the 'original' non-truncated MDP.

Clearly, Φ is not a product set, essentially because the truncation parameter can not be state-dependently chosen. To see this, let $\phi^1 = (N_1, N_2, \delta)$ and $\phi^2 = (N_1', N_2', \delta')$, with $(N_1', N_2') \neq (N_1, N_2)$. There is no state i for which there exists $\phi \in \Phi$ with property (5.1).

Finally, the verification of Assumption 1 is trivial. Note further that $\mathcal{A}(i)$ is compact (in the discrete topology). Also, $\mathbf{Z}_+ \cup \{\infty\}$ is compact in the one-point compactification, based on the discrete topology. By Tychonov's theorem, the infinite product of compact spaces is compact in the product topology, and so Φ is compact.

Since convergence in the product topology is the topology of pointwise convergence, the desired continuity properties then immediately follow.

Next we define the various performance measures and optimality criteria that we will study. Later on we will provide conditions under which these are well-defined, and optimal polices exist.

For $0 < \alpha < 1$, define the *expected total α-discounted cost value function* V_α^ϕ under parameter $\phi \in \Phi$ by

$$V_\alpha^\phi(i) = \mathsf{E}_i^\phi \left[\sum_{n=0}^{\infty} (1-\alpha)^n c(X_n) \right], \quad i \in S. \tag{5.2}$$

Notice that the discount factor is taken to be $1-\alpha$ to allow for a more direct analogy to the continuous time case.

Next suppose that Φ has the product property. Define the *minimum expected total α-discounted cost* V_α^* in Φ by

$$V_\alpha^*(i) = \inf_{\phi \in \Phi} V_\alpha^\phi(i), \quad i \in S.$$

If for some $\phi \in \Phi$ it holds that $V_\alpha^\phi = V_\alpha^*$, then ϕ is said to be an α-discount optimal policy (in Φ).

The *expected average cost* g^ϕ under parameter $\phi \in \Phi$ is given by

$$\mathrm{g}^\phi(i) = \limsup_{N \to \infty} \frac{1}{N+1} \mathsf{E}_i^\phi \left[\sum_{n=0}^{N} c(X_n) \right], \quad i \in S.$$

If Φ has the product property, the *minimum expected average cost* (w.r.t. Φ) is defined as

$$\mathrm{g}^*(i) = \inf_{\phi \in \Phi} \mathrm{g}^\phi(i), \quad i \in S.$$

If for $\phi \in \Phi$ it holds that $\mathrm{g}^\phi = \mathrm{g}^*$, then ϕ is called an average optimal policy (in Φ).

A stronger notion of optimality, called Blackwell optimality, applies more often than is generally noted. We define it next (see also [17]).

Let Φ have the product property. The policy $\phi^* \in \Phi$ is *Blackwell optimal* in Φ, if for any $i \in S$, $\phi \in \Phi$, there exists $\alpha(i, \phi) > 0$, such that $V_\alpha^{\phi^*}(i) \leq V_\alpha^\phi(i)$ for $\alpha < \alpha(i, \phi)$. Additionally, ϕ^* is *strongly Blackwell optimal*, if $\inf_{i \in S, \phi \in \Phi} \alpha(i, \phi) > 0$.

The restriction to a product set in the above optimality criteria is due to the fact that these criteria typically involve per state minimisations. Without the product property, the existence of a parameter that is a minimiser for all states simultaneously would be a mere coincidence.

5.2.1 Discounted Cost

To determine the discounted cost V_α, an important instrument is the *(discrete time) discount optimality equation* (DDOE)

$$u(i) = \inf_{\phi \in \Phi} \left\{ c^\phi(i) + (1-\alpha) \sum_{j \in S} p^\phi(j\,|\,i) u_j \right\}, \quad i \in S, \tag{5.3}$$

w.r.t. to a parameter set Φ with the product property. In this subsection we show that mild conditions guarantee the existence of a unique solution to this equation in a certain space of functions. Moreover, the inf is a min, and a minimising policy in (5.3) is optimal in Φ (and even optimal within the larger set of randomised and non-stationary policies generated by Φ).

The condition used here has been taken from [34, 54].

Definition 5.2. Let $\gamma \in \Re$. The function $M : S \to (0,\infty)$ is called a (γ, Φ)-drift function if $P^\phi M \le \gamma M$ for all $\phi \in \Phi$. Note that '$\le$' stands for component-wise ordering.

Definition 5.3. The Banach space of M-bounded functions on S is denoted by $\ell^\infty(S,M)$. This means that $f \in \ell^\infty(S,M)$ if and only if $f : S \to \Re$ and

$$\|f\|_M = \sup_{i \in S} \frac{|f(i)|}{M(i)} < \infty.$$

Assumption 2 (α)

1. *There exist a constant* $\gamma < 1/(1-\alpha)$ *and a function* $M : S \to (0,\infty)$ *such that* M *is* (γ, Φ)*-drift function and that* $\phi \mapsto P^\phi M$ *is component-wise continuous;*
2. $b_M := \sup_\phi \|c^\phi\|_M < \infty$.

The above assumption allows to rewrite (5.2) as

$$V_\alpha^\phi = \sum_{n=0}^{\infty} (1-\alpha)^n P^{\phi,(n)} c^\phi.$$

The following lemma is quite straightforward to prove. For completeness we give the details.

Lemma 5.1. *Suppose that the Assumptions 1 and 2* (α) *hold, then* $\phi \mapsto V_\alpha^\phi$ *is component-wise continuous and* V_α^ϕ *is the unique solution in* $\ell^\infty(S,M)$ *to*

$$u = c^\phi + (1-\alpha) P^\phi u. \tag{5.4}$$

Proof. First notice that $V_\alpha^\phi \in \ell^\infty(S,M)$, since

$$|V_\alpha^\phi| = |\sum_{n=0}^{\infty}(1-\alpha)^n P^{\phi,(n)}c^\phi| \le \sum_{n=0}^{\infty}(1-\alpha)^n P^{\phi,(n)} b_M \cdot M$$
$$\le \sum_{n=0}^{\infty}(1-\alpha)^n \gamma^n b_M \cdot M = \frac{b_M}{1-(1-\alpha)\gamma} M.$$

Next, V_α^ϕ is a solution to Eq. (5.4), since

$$(1-\alpha)P^\phi V_\alpha^\phi = (1-\alpha)P^\phi \sum_{n=0}^{\infty}(1-\alpha)^n P^{\phi,(n)}c^\phi$$
$$= \sum_{n=1}^{\infty}(1-\alpha)^n P^{\phi,(n)}c^\phi = V_\alpha^\phi - c^\phi.$$

Let $f = (f(i))_i \in \ell^\infty(S,M)$ be any solution to Eq. (5.4), then

$$V_\alpha^\phi(i) - f(i) = (1-\alpha)\sum_j p^\phi(j\,|\,i)V_\alpha^\phi(j) - f(j))$$
$$= (1-\alpha)^n \sum_j p^{\phi,(n)}(j\,|\,i)(V_\alpha^\phi(j) - f(j)).$$

Hence,

$$|V_\alpha^\phi(i) - f(i)| \le (1-\alpha)^n \sum_j p^{\phi,(n)}(j\,|\,i)|V_\alpha^\phi(j) - f(j)|$$
$$\le (1-\alpha)^n P^{\phi,(n)}(\phi)M(j) \cdot \big(\frac{b_M}{1-(1-\alpha)\gamma} + \|f\|_M\big)$$
$$\le (1-\alpha)^n \gamma^n M(i) \cdot \big(\frac{b_M}{1-(1-\alpha)\gamma} + \|f\|_M\big) \to 0, \quad n \to \infty.$$

This implies $f = V_\alpha^\phi$, hence V_α^ϕ is the unique solution to Eq. (5.4) in $\ell^\infty(S,M)$.

Finally, to show that $\phi \mapsto V_\alpha^\phi(i)$, $i \in S$, is continuous, notice that by assumption $\phi \mapsto P^\phi M$ is component-wise continuous. It follows that $\phi \mapsto P^{\phi,(n)}M$ component-wise continuous. Since $P^{\phi,(n)}M \le \gamma^n M$, the dominated convergence theorem yields that $\phi \mapsto \sum_{n=0}^\infty (1-\alpha)^n P^{\phi,(n)}M < \infty$ is component-wise continuous. Furthermore, since $\phi \mapsto c^\phi$ is component-wise continuous and $|c^\phi| \le b_M \cdot M$, an application of the generalised dominated convergence theorem ([41, Proposition 11.18]) implies component-wise continuity of $\phi \mapsto \sum_{n=0}^\infty (1-\alpha)^n P^{\phi,(n)}c^\phi = V_\alpha^\phi$.

The following theorem is a well-known result by Wessels [54].

Theorem 5.1 (cf. Wessels [54]). *Suppose that Φ is a compact set with the product property, and suppose that Assumptions 1 and 2(α) hold. Then V_α^* is finite and the unique solution in $\ell^\infty(S,M)$ to the discount optimality equation (DDOE) (5.3) w.r.t. Φ.*

Moreover, the infimum is attained as a minimum. For any $\phi^* \in \Phi$, for which ϕ^* achieves the minimum in Eq. (5.3) for all $i \in S$, it holds that $V_\alpha^{\phi^*} = V_\alpha^*$ and ϕ^* is (α-discount) optimal in Φ.

The versatile applicability of (γ, Φ)-drift functions is illustrated in [9, 14], and the examples below. First note for the bounded cost case, that the function $M \equiv 1$ is an appropriate function satisfying Assumption 2(α).

We give two examples, one without and the second with truncation.

Example 3 (Quality of Service) *Consider a discrete time single server queue, where the probability of an arrival in the next time slot is $p \in (0,1)$. The system state represents the number of customers in the system, hence, the state space is $S = \{0, 1, \dots\}$. The probability of a service completion in the next time slot is subject to control, and can be chosen from the set $[\mu_l, \mu_h] \subset [0,1]$, depending on the system state.*

The system manager incurs a holding cost h per unit time, per customer in the system. Furthermore, operating the system with service completion probability $\mu \in [\mu_l, \mu_h] \subset [0,1]$ induces cost $c(\mu)$ per unit time, where $\mu \mapsto c(\mu)$ is a continuous, increasing function. The objective is to select the service probability per state so as to minimise the overall cost: selecting a high service probability induces a larger decrease of the number of customers in the system and hence of the total holding cost per unit time. On the other hand, the service quality cost is also higher when operating at a high service level. Thus, the two types of cost should be balanced in an optimal service probability assignment rule.

We assume that arrivals and service completions occur independently of each other. As a consequence, a simultaneous arrival and service completion may occur within one time-slot.

The per state control is the selection of a service completion probability, provided the system state does not equal 0. Therefore, put $\Phi = \mathcal{A}$, where $\mathcal{A}(i) = [\mu_l, \mu_h]$, $i \neq 0$, $\mathcal{A}(0) = \{0\}$, and $\mathcal{A} = \prod_i \mathcal{A}(i)$. To any fixed parameter $\phi = (\mu_0 = 0, \mu_1, \mu_2, \dots)$, one may associate a Markov process, representing the system state at any time t, with transition probabilities given by

$$p^\phi(j \mid i) = \begin{cases} p(1-\mu_i), & j = i+1 \\ (1-p)\mu_i, & j = (i-1)\mathbf{1}_{\{x \neq 0\}} \\ 1-p-\mu_i+2p\mu_i, & j = i. \end{cases}$$

The corresponding cost vector is given by

$$c^\phi(i) = c(\mu_i) + hi,$$

and $c^\phi(0) = 0$. Assumption 1 is directly verified.

As an appropriate (γ, Φ)-drift function, we may choose $M(i) = e^{\varepsilon i}$, with $\varepsilon > 0$ to be determined below:

$$\sum_j p^\phi(j \mid i) e^{\varepsilon j} = (1-p)\mu_i e^{\varepsilon(i-1)} + (1-(1-p)\mu_i - p(1-\phi_i))e^{\varepsilon i}$$

$$
\begin{aligned}
&+p(1-\mu_i)e^{\varepsilon(i-1)} \\
&= e^{\varepsilon i}\Big(1+p(1-\mu_i)(e^{\varepsilon}-1)+(1-p)\mu_i(1-e^{-\varepsilon})\Big) \\
&\leq e^{\varepsilon i}\big(1+p(e^{\varepsilon}-1)\big).
\end{aligned}
$$

For $\varepsilon = 0$, the coefficient of $e^{\varepsilon i}$ in the above equals 1. Since $1/(1-\alpha) > 1$, one can always choose ε small enough so that

$$\gamma := 1 + p(e^{\varepsilon} - 1) < \frac{1}{1-\alpha}.$$

As a consequence, Assumption 2(α),(i) is satisfied.

We check Assumption 2(α),(ii):

$$b_M := \sup_{\phi \in \Phi} \sup_i \frac{|c^{\phi}(i)|}{M(i)} \leq \sup_i \frac{c(\mu_h) + hi}{M(i)} < \infty. \tag{5.5}$$

The example shows, that the existence of a (γ, Φ)-drift function does not impose any restrictions on the class structure of the associated Markov processes and transience is allowed as well. Moreover, it is often a good and simply checked choice to take V exponential. Since generally cost structures are linear or quadratic as a function of state, they are dominated by exponential functions. Thus, they fit in the framework discussed here. We are not aware of a specific interpretation of the function M.

Example 4 (Continuation of Examples 1, and 2 Server Allocation) *The formal description of the model has been provided in Example 2. As in the previous example, it is straightforward to check that for each $\alpha > 0$, one can determine ε_{i_1} and $\varepsilon_{i_2} > 0$, such that*

$$M(i) = e^{\varepsilon_{i_1} i_1 + \varepsilon_{i_2} i_2}, \quad i \in S, \tag{5.6}$$

is a (γ, Φ)-drift function for $\gamma < 1/(1-\alpha)$. Since c^{ϕ} is a linear function of the state, and M is exponential, also (5.5) is satisfied. This implies that Assumption 2 (α) is satisfied.

The verification of Assumption 1 is direct, since the support of the transition probabilities is finite, and elementwise continuity has been checked in Example 2. Hence, the results of Theorem 5.1 apply for Φ_{N_1,N_2}, that is for the MDP with the fixed truncation parameter (N_1, N_2). Furthermore, the results of Lemma 5.1 hold. Thus, denoting the α-discount optimal value function w.r.t. Φ_{N_1,N_2} by $V_{\alpha}^{*,(N_1,N_2)}$, and the optimal policy by $(N_1, N_2, \delta^*_{N_1,N_2})$, it holds that $V_{\alpha}^{*,(N_1,N_2)} \to V_{\alpha}^{*,(\infty,\infty)}$, $N_1, N_2 \to \infty$ and any limit point of the sequence $(N_1, N_2, \delta^*_{N_1,N_2})$ is α-discount optimal in $\Phi_{\infty,\infty}$. Hence, it is α-discount optimal for the non-truncated MDP.

5.2.1.1 Value Iteration

A very important algorithm to approximate V_α is the value iteration algorithm (VI), originally due to Bellman [5]. Here we assume that Φ has the product property.

Algorithm 1 VI for an α-discounted cost ε-optimal policy

1. Select $V_{\alpha,0} \in \ell^\infty(S,M)$, specify $\varepsilon > 0$, set $n = 0$.
2. For each $i \in S$, compute $V_{\alpha,n+1}(i)$ by

$$V_{\alpha,n+1}(i) = \min_{\phi\in\Phi}\Big\{c^\phi(i) + (1-\alpha)\sum_{j\in S} p^\phi(j\,|\,i)V_{\alpha,n}(j)\Big\}, \tag{5.7}$$

   and let

$$\phi_{n+1} \in \arg\min_{\phi\in\Phi}\{c^\phi + (1-\alpha)\sum_{j\in S} P^\phi V_{\alpha,n}\}.$$

3. If

$$\|V_{\alpha,n+1} - V_{\alpha,n}\|_V \le \frac{1-(1-\alpha)\gamma}{2(1-\alpha)\gamma}\varepsilon,$$

   then put $V^\varepsilon := V_{\alpha,n+1}$, $\phi^\varepsilon := \phi_{n+1}$, stop. Otherwise increment n by 1 and return to step 2.

Theorem 5.2 (cf. [54], [39, Theorem 6.3.1]). *Suppose that Φ is a compact set with the product property, and suppose that Assumptions 1 and 2(α) hold. Let $V_{\alpha,0} \in \ell^\infty(S,M)$ and $\varepsilon > 0$. Let $\{V_{\alpha,n}\}_{n\in\mathbb{N}}$ satisfy Eq. (5.7) for $n \ge 1$. Then the following hold.*

i) $\lim_{n\to\infty}\|V_\alpha^* - V_{\alpha,n}\|_M = 0$, *in particular,*

$$\begin{aligned}\|V_\alpha^* - V_{\alpha,n}\|_M &\le \frac{1}{1-(1-\alpha)\gamma}\|V_{\alpha,n+1} - V_{\alpha,n}\|_M \\ &\le \frac{(1-\alpha)^n\gamma^n}{1-(1-\alpha)\gamma}\|V_{\alpha,1} - V_{\alpha,0}\|_M.\end{aligned}$$

Any limit point of the sequence $\{\phi_n\}_n$ is an α-discount optimal policy (in Φ).

ii) $V^\varepsilon = V_{\alpha,n+1}$ *is an $\varepsilon/2$-approximation of V_α^*, in other words,* $\|V_\alpha^* - V_{\alpha,n+1}\|_M \le \frac{\varepsilon}{2}$.

iii) $\phi^\varepsilon = \phi_{n+1}$ *is an ε-optimal policy, in other words,* $\|V_\alpha^* - V_\alpha^{\phi^\varepsilon}\|_M \le \varepsilon$.

Proof. The proof of Theorem 5.2 (i) is straightforward using that

$$V_\alpha^* - V_{\alpha,n} = \lim_{m\to\infty}\sum_{k=n}^{m}(V_{\alpha,k+1} - V_{\alpha,k}).$$

The bounds in (ii, iii) are given somewhat implicit in [54]. Their derivation is completely analogous to the derivation of the bounds of e.g. [39, Theorem 6.3.1] for the bounded reward case, with λ replaced by $(1-\alpha)\gamma$. □

Example 5 (Continuation of Examples 1, 2, and 4 Server Allocation) *For each truncation parameter (N_1, N_2), the corresponding parameter set is compact. Since the assumptions of Theorem 5.2 are satisfied (cf. Example 4), VI converges for any fixed truncation parameter (N_1, N_2).*

Remark 5.2. The reader may wish to point out that a solution to the DDOE yielding an α-discount deterministic policy exists without any further conditions in the case of non-negative cost (negative dynamic programming, cf. [48]) and a finite action space per state. Also VI converges provided $v_0 \equiv 0$, although no convergence bounds can be provided. If the action space is compact, additional continuity and inf-compactness (cf. [24, Corollary 5.7]) properties are necessary for the existence of a stationary deterministic policy attaining the minimum in the DDOE. It is not clear to us how these conditions could be extended in order to include parametrised Markov processes.

Notice further, that, unfortunately, in general there is no unique solution to the DDOE (cf. [24], [44, Sect. 4.2]). Using norm conditions as in this chapter, allows to identify the value function as the unique one in the Banach space of functions bounded by M (cf. Theorem 5.1). In the non-negative cost case, the value function is the minimum solution to the DDOE (see [44, Theorem 4.1.4]).

In case of a finite state space, VI can be numerically implemented. In the case of a countable space, its use is restricted to the derivation of structural properties of the value function and α-discount optimal policy. Structural properties such as non-decreasingness, convexity, etc. can be used to show for instance that a threshold policy or an index policy is optimal. The existence of an optimal policy with desired properties can be directly derived from the structure of the value function V_α^* in combination with the DDOE. Alternatively, this can be deduced from the fact that each ϕ^n having the desired properties implies that any limit point has.

We summarise the procedure via a roadmap.

5.2.1.2 Roadmap to Structural Properties

Below we formulate a scheme for deriving the structure of an optimal policy and value function, if the optimisation criterion is to minimise the expected α-discounted cost. Let $\Phi = \mathcal{A}$ be the (product) set of all stationary, deterministic policies.

Roadmap for α-discounted cost MDPs in discrete time

1. Check the conditions of Theorem 5.1.
   If satisfied then:

- perform VI Algorithm 1, and check iteratively that the structural properties of interest hold.

2. If not satisfied, or if no structural properties can be iteratively concluded, then:
   - the outcome is inconclusive.

A main reference on the propagation of structural properties through the VI induction step Eq. (5.7) is [33]. The technique discussed in this monograph is called *event based dynamic programming*, and it presents a systematic framework of propagations of the desired structural properties for operators that represent events. We have developed new operators in [8, 12, 14], for special perturbations or truncations that tend to leave the structural properties of optimal policies in the original model intact. Below we present an example. A further application to MDPs with unbounded jump rates will be discussed in Sect. 5.3.4.

5.2.2 Approximations/Perturbations

As indicated in the above, we will next focus our attention to parameters capturing a perturbation of the MDP. This parameter set should parametrise the collection of deterministic policies $\mathcal{A}$, as well as a perturbation set $\mathcal{N}$. This perturbation can have multiple interpretations, depending on the context. It can be a finite state approximation, or it can represent some uncertainty in the input parameters. Put $\Phi = \mathcal{N} \times \mathcal{A}$. As has been illustrated in Example 2, Φ need not have the product property. However, fixing the perturbation parameter at $N \in \mathcal{N}$, the associated set of parameters $\{N\} \times \mathcal{A}$ generally does. Denote by $V_\alpha^{*,(N)}$ the optimal α-discount value function with respect to parameter set $N \times \mathcal{A}$, i.o.w. the optimal α-discount value function for the N-perturbation.

The following continuity result follows directly from Lemma 5.1.

Corollary 5.1 (To Lemma 5.1 and Theorem 5.1). *Suppose that $\Phi = \mathcal{N} \times \mathcal{A}$, where $\{N\} \times \mathcal{A}$ has the product property for any $N \in \mathcal{N}$, and that Assumptions 1, and 2 (α) hold. Then,*

1. $\lim_{N \to N_0} V_\alpha^{*,(N)} = V_\alpha^{*,(N_0)}$, *where $V_\alpha^{*,(N)}$ is the unique solution of the DDOE (5.3) in $\ell^\infty(S,M)$ with respect to $\{N\} \times \mathcal{A}$, $N \in \mathcal{N}$;*
2. *for $\phi^*(N) = \{N, \delta^*(N))\}$ α-discount optimal in $\{N\} \times \mathcal{A}$, it holds that any limit point of the sequence $\{\phi^*(N)\}_{N \to N_0}$ is α-discount optimal in $\{N_0\} \times \mathcal{A}$.*

Without the existence of a (γ, Φ)-drift function bounding the one-step cost uniformly in the parameter, the above convergence result may fail to hold.

Example 6 *(cf. [44, Example 4.6.1]) Let the parameter set $\Phi = \{3, \ldots\} \cup \{\infty\}$, $S = \{0, 1, \ldots\}$. In this case, there is only a perturbation parameter that we denote by $N \in \Phi$ (and no control).*

The transition probabilities are as follows.

$$p^{\infty}(j\,|\,i) = \frac{1}{2} \quad j \in \{0, i+1\},$$

and for $N < \infty$

$$p^{N}(j\,|\,i) = \begin{cases} \frac{1}{2}, & i \neq N-1, N,\ j = 0 \\ \frac{1}{2} - \frac{1}{N}, & i \neq N-1, N,\ j = i+1 \\ \frac{1}{N}, & x \neq N,\ j = N \\ 1 - \frac{1}{N}, & i = N-1,\ j = 0 \\ 1, & x = y = N. \end{cases}$$

Further let $\alpha > \frac{1}{2}$ *and define* $c^{N}(i) = i^2$ *for* $N \leq \infty$. *The calculations in [44, Example 4.6.1] show that* $V_{\alpha}^{(\infty)}(0) < \lim_{N\to\infty} V_{\alpha}^{(N)}(0) = \infty$. *It is simply checked that any* (γ, Φ)*-drift function* M *can at most be linear. Indeed for* $3 \leq N < \infty$, *it must hold that*

$$\frac{1}{2}M(0) + \Big(\frac{1}{2} - \frac{1}{N}\Big)M(2) + \frac{1}{N}M(N) \leq \gamma M(1),$$

leading to the requirement that $M(N) \leq N\gamma M(1)$, *hence* $\sup_{N,i} \frac{c^{N}(i)}{M(i)} = \infty$.

Literature related to Corollary 5.1 can be found in [39, Sect. 6.10.2], [44, Sect. 4.6] and [37]. The first two papers consider finite space truncations. The first reference further assumes the existence of a (γ, Φ)-drift function without continuity, but with an additional tail condition and a prescribed truncation method. These conditions are implied by ours.

The second reference [44, Sect. 4.6] considers minimisation of non-negative costs, as well as a finite action space per state. The truncation method developed is called the 'approximating sequence' method. If there is a uniform bound on the costs, no assumptions have to be made, other than that the truncation is a finite state truncation, that the decision sets are independent of the truncation, and that the transition probabilities are continuous in the truncation parameter. If the costs are allowed to be unbounded as a function of state, then conditions on the truncation method have to be made.

Finally, the paper that is closest related to the setup presented in the present paper is [37]. The analogous statement to Corollary 5.1 is [37, Theorem 3.1]. The conditions imposed are basically the same as the ones presented here, though (slightly) more restrictive (cf. [13, Remark 5.2]). Additionally, the approach in [37] is not a parametrised one, but focusses on perturbations of MDPs.

The issue of error bounds is not addressed in these works. An interesting reference on perturbations of Markov processes with error bounds is [52].

Example 7 (Continuation of Example 6) *The above example is a case where neither the conditions on convergence of the approximating sequence method from [44] are met, nor the ones presented in this paper.*

However, the conditions in the approximating sequence method of [44] are not even fulfilled, if we change the cost function to $c^{N}(i) = i$ *for all* $N \leq \infty$, $i \in S$. *On*

the other hand, the function M defined by $M(i) = i$, $i \in S$, is a (γ, Φ)-drift function, for which the conditions of Corollary 5.1 are trivially met. Hence, the set-up in this paper can be applied and we find that $\lim_{N\to\infty} V_\alpha^N(0) = V_\alpha^\infty(0) < \infty$.

5.2.2.1 Type of Perturbations

Although any perturbation satisfying the conditions of Corollary 5.1 yields (component-wise) continuity of the value function as a function the perturbation parameter, not any perturbation is desirable in terms of structural properties.

To explain this, we consider again the server allocation Examples 1, 2, 4 and 5.

Example 8 (Continuation of Examples 1, 2, 4, and 5 Server Allocation) *Assume that $(1-p_1)\mu_1 > \mu_2$, and fix the truncation parameter at $N_1 = N_2 = \infty$, i.o.w., we consider the unperturbed MDP with parameter space $\Phi_{\infty,\infty} = \{(\infty,\infty)\} \times \prod_i \mathcal{A}(i)$.*

By using VI, event based dynamic programming yields that allocating the server to unit 1, when non-empty, is α-discount optimal (cf. [12, Chap. 6]):

$$\delta^*(i) = \begin{cases} 1, & i_1 \neq 0 \\ 2, & i_1 = 0, i_2 \neq 0 \\ 0, & i_1 = i_2 = 0. \end{cases} \tag{5.8}$$

Thus, $\phi^* = (\infty, \infty, \delta^*)$ is α-discount optimal in $\Phi_{\infty,\infty}$. Indeed, allocation to unit 1 gives a larger cost reduction per unit time due to customer service completions than allocating to unit 2. Thus, noticing that the cost rate per unit time and per server unit are equal to 1, this allocation policy is a generalised $c\mu$-rule. Let us refer to this policy as AP1 (allocation to unit 1 policy). Since this is true for any $0 < \alpha < 1$, AP1 is strongly Blackwell optimal.

As has been deduced in Example 4, for any truncation parameter (N_1, N_2), there exists a (stationary, deterministic) optimal policy $(N_1, N_2, \delta^*_{N_1,N_2})$ in Φ_{N_1,N_2} and any limit point of the sequence $(N_1, N_2, \delta^*_{N_1,N_2})$, $N_1, N_2 \to \infty$ is optimal in $\Phi_{\infty,\infty}$. Now, we choose the following parameters $\lambda = 1/9.84$, $\mu_1 = 8.56/9.84$, $\mu_2 = 0.28/9.84$, $p_1 = 0.22$.

Setting $N_1 = N_2 = 300$ leads to the optimal policy in the rectangle shown in the picture below.

This optimal policy is very far from being the index policy AP1, although the truncation size seems large enough to exhibit an optimal policy that is 'closer' to AP1. One starts to wonder what the effect of such a straightforward truncation has been on numerical approximations of other models studied in the literature.

The question is, whether there is a generic truncation method that does not have this defect.

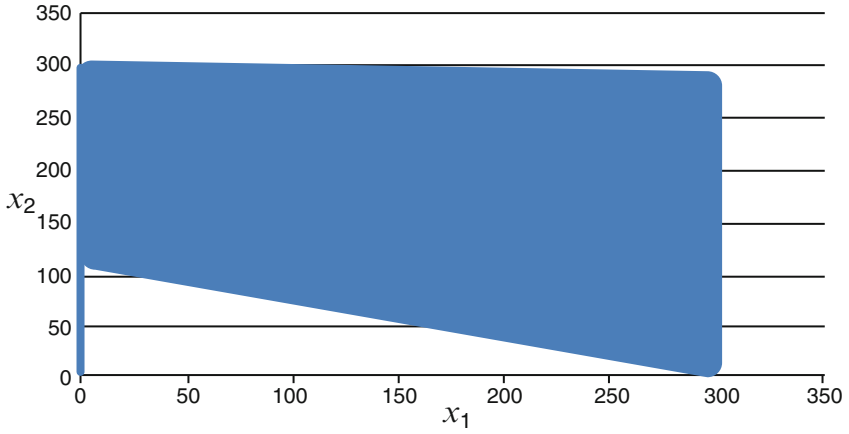


Fig. 5.2: Standard Truncation with $N_1 = N_2 = 300$: optimal allocation to unit 2 in *blue* states

5.2.2.2 Smoothed Rate Truncation (SRT)

SRT is a perturbation introduced in [8], where 'outward bound' probabilities are linearly decreased as a function of state. This creates a perturbed MDP with a finite closed class under any policy. The idea is best illustrated by specifying possible SRTs for the above example. The experience with SRT so far is that it leaves the structure of an optimal policy intact (cf. [8, 9]). On the other hand, since it perturbs transition probabilities from *all* states in the finite closed class, the value function itself seems better approximated by a straightforward cut-off, such as the one as described above.

Example 9 (Continuation of Examples 1, 2, 4, 5 and 8 Server Allocation) *One can apply SRT as follows. Let Φ be the parameter set from Example 2. For $\phi \in \Phi$ with $\phi = (N_1, N_2, \delta)$ we now define*

$$p^{\phi}(j\,|\,i) = \begin{cases} \lambda(1 - i_1/N_1)^+ & j = (i_1+1, i_2) \\ p_1\mu_1\big(1 - i_2/N_2\big)^+ & j = (i_1-1, i_2+1), i_1 > 0, \delta(i) = 1 \\ (1-p_1)\mu_1 + & \\ \quad \mathbf{1}_{\{i_2 \le N_2\}} p_1\mu \cdot i_2/N_2 & j = (i_1-1, i_2), i_1 > 0, \delta(i) = 1 \\ \mu_2 & j = (i_1, i_2-1), i_2 > 0, \delta(i) = 2 \\ 1 - \sum\limits_{w \neq i} p^{\phi}(j\,|\,i) & j = i. \end{cases}$$

Again, the function M from (5.6) is a (γ, Φ)-drift function for some $\gamma > 0$, satisfying Assumptions 1 and 2(α).

The following picture illustrates the numerical results with $N_1 = N_2 = 35$. This confirms the results in [8, 9], suggesting that SRT allows to obtain information on the structure of an optimal policy for an infinite state MDP. Notice, that smoothly truncating at size $N_1 = N_2 = 35$ seems far more restrictive than truncating at $N_1 = N_2 = 300$ as in Example 8. However, the optimal policy is AP1, which is optimal for the non-truncated MDP.

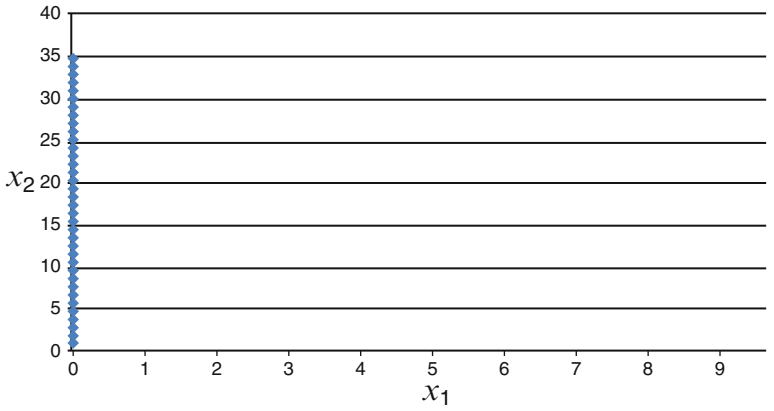


Fig. 5.3: Smoothed Rate Truncation with $N_1 = N_2 = 35$: optimal allocation to unit 2 in *blue* states

Numerical results show that AP1 is optimal in both the truncated and the non-truncated MDPs. We have not proven this result for the rectangular SRT, but we did so for another SRT (cf. [12, Chap. 6]) that is not based on a rectangle but on a triangle. The SRT is as follows. Put

$$\Phi = \{\phi \,|\, \phi = (N, \delta) \,|\, N \in \mathbf{Z}_+ \cup \{\infty\}, \delta \in \mathcal{D}\}.$$

Fix N. Then we put

$$p^{(N,\delta)}(j\,|\,i) = \begin{cases} \lambda(1-(i_1+i_2)/N)^+, & j = (i_1+1, i_2) \\ p_1\mu_1 & i_1 > 0, j = (i_1-1, i_2+1), \delta(i) = 1 \\ (1-p_1)\mu_1 & i_1 > 0, y = (i_1-1, i_2), \delta(i) = 1 \\ \mu_2 & i_2 > 0, y = (i_1, i_2-1), \delta(i) = 2 \\ 1 - \sum_{w \neq i} p^{(N,\delta)}(w\,|\,i) & j = i. \end{cases}$$

Using a triangular truncation requires less events to be truncated and so the proofs are simpler. This illustrates that smooth truncations are not uniquely defined, but different choices may be possible.

5.2.3 Average Cost

Establishing a framework for the average cost optimality criterion is more difficult than for the discounted cost case. There are several cautionary examples in the literature highlighting the complications. In our opinion, the most intricate one is the Fisher-Ross example [25]. In this example, all action spaces are finite, the Markov process associated with any stationary deterministic policy is irreducible and has a stationary distribution. However, there is an optimal non-stationary policy, but no stationary, deterministic one is optimal.

In this chapter we provide conditions under which there exists a stationary average optimal policy satisfying the average cost optimality equation. The proof requires the vanishing discount approach. Our focus in this paper are non-negative cost functions, the analysis of which does not require a heavy drift condition that imposes positive recurrence of the Markov process associated with any parameter. However, below, we do touch upon benefits of using the heavier conditions.

The conditions that we will focus on, are based on the work of Borkar [15], in the form discussed in [44]. They resemble the conditions from the earlier work in [53]. They imply conditions developed by Sennott in [42], requiring (i) lower-bounded direct cost; (ii) the existence of a finite α-discounted value function, (iii) the existence of a constant L and a function $h : S \to \Re$, such that $-L \leq V_\alpha^*(i) - V_\alpha^*(z) \leq h(i)$, $i \in S$, for some state z, and all α sufficiently small, and (iv) there exists ϕ, such that $\sum_j p^\phi(j\,|\,i)h(j) < \infty$. Under these conditions Sennott [42] proves the statement in Theorem 5.3 below, with equality in (5.9) replaced by inequality, and without property (3) from Theorem 5.3. It is appropriate to point out that the approach initiated in [42] was based on a bounded cost average optimality result in [40].

The following definitions are useful. Define $\tau_z := \min\{n \geq 1\,|\,X_n = z\}$ to denote the first entrance time of $z \in S$. Let $\mu_z^\phi(i) = \mathsf{E}_i^\phi\left[\tau_z\right]$ and $C_z^\phi(i) = \mathsf{E}_i^\phi\left[\sum_{n=0}^{\tau_z} c(X_n)\right]$. One may replace a single state z by a set, and the analogous definitions prevail.

Assumption 3 *Suppose that Φ has the product property. The following holds.*

1. *Non-negative cost rates: $c^\phi(i) \geq 0$ for all $i \in S, \phi \in \Phi$.*
2. *There exist $z \in S$ and $\phi_0 \in \Phi$ such that $\mu_z^{\phi_0}(i), C_z^{\phi_0}(i) < \infty$ for all $i \in S$. Note that this implies that $\mathsf{g}^{\phi_0}(i)$ is independent of $i \in S$ and hence we write $\mathsf{g}^{\phi_0} = \mathsf{g}^{\phi_0}(i)$, for all $i \in S$.*
3. *There exists $\varepsilon > 0$ such that $D = \{i \in S\,|\,c^\phi(i) \leq \mathsf{g}^{\phi_0} + \varepsilon$ for some $\phi \in \Phi\}$ is a finite set.*
4. *For all $i \in D$, there exists $\phi^i \in \Phi$ such that $\mu_i^{\phi^i}(z), C_i^{\phi^i}(z) < \infty$.*

Notice, that by virtue of Assumption 3, the Markov process operated under parameter ϕ_0 has one positive recurrent class containing z, with finite expected average cost $\mathsf{g}(\phi_0)$. The expected time and total expected cost incurred until reaching that class are finite, for any initial state.

Theorem 5.3. *Suppose that Φ is a compact set with the product property, and that Assumptions 1, 2 (α), and 3 hold, for all $\alpha \in (0,1)$. Then the following holds.*

i) There exists a solution tuple (g^*, H^*), $g^* \in \mathfrak{R}_+$, $H^* : S \mapsto \mathfrak{R}$, *to the average cost optimality equation (DAOE)*

$$g + u(i) = \min_{\phi \in \Phi} \Big\{ c^{\phi}(i) + \sum_{j \in S} p^{\phi}(j \,|\, i) u(j) \Big\}, \tag{5.9}$$

with the properties that

(1) $g^* = \mathrm{g}^*$ *is the minimum expected average cost (in* Φ*),*
(2) any $\phi^* \in \Phi$ *with*

$$\phi^* \in \arg\min_{\phi \in \Phi} \Big\{ c^{\phi} + P^{\phi} H^* \Big\},$$

is (average cost) optimal in Φ *and*
(3) there exists $i^* \in D$ *with* $H^*(i^*) = \inf_i H^*(i)$.

ii) Let $i_0 \in S$. *Any sequence* $\{\alpha_n\}_n$ *with* $\lim_n \alpha_n = 0$, *has a subsequence, again denoted* $\{\alpha_n\}_n$, *along which the following limits exist:*

$$\begin{aligned} H'(i) &= \lim_{n \to \infty} \big(V^*_{\alpha_n}(i) - V^*_{\alpha_n}(i_0) \big), \quad i \in S, \\ g' &= \lim_{n \to \infty} \alpha_n V^*_{\alpha_n}(i), \quad i \in S \\ \phi' &= \lim_{n \to \infty} \phi^{\alpha_n}, \end{aligned}$$

with ϕ^{α_n} α_n*-discount optimal in* Φ. *Any such tuple* (g', H') *is a solution to Eq. (5.9) with the properties (1), (2) and (3), so that* $g' = \mathrm{g}^*$. *Moreover,* ϕ' *takes minimising actions in Eq. (5.9) for* $g = g'$ *and* $u = H'$.

Theorem 5.3 is a slight extension from [44, Theorem 7.5.6], where the action space is assumed to be finite. Although the various proof parts are scattered over [44, Chap. 7], we will merely indicate the necessary adjustments to allow for the compact parameter case. We would like to note that we use a completely analogous reasoning in the proof of Theorem 5.7, which contains all further details.

Proof. A close examination of the proof of [44, Theorem 7.5.6] shows that the assumption of a finite action space is not necessary. The proof can be adjusted in such a way that the statement holds for a compact action space as well. We briefly discuss the adjustments below. The existence of the limits along a sequence $\{\alpha_n\}_n$, $\alpha_n \downarrow 0$, $n \to \infty$, in assertion (ii) is a direct result of Sennott [44].

Obtaining the average cost optimality inequality (DAOI) for a limit point of α-discount optimal policies, as $\alpha \to 0$, can be achieved by virtue of Fatou's lemma. This policy is shown to be optimal.

Further, one needs to show explicitly that there exists a policy realising the infimum of Eq. (5.9). Since the limit policy satisfies the DAOI, a similar reasoning as in the (very ingenious) proof of Sennott [44, Theorem 7.4.3] yields that this policy satisfies the DAOE as well. In fact, any policy satisfying the DAOI also satisfies the DAOE. It can then be shown by contradiction that this limit policy must attain

the infimum. As a consequence, the limit tuple (g', H') from (ii) is a solution to Eq. (5.9). The rest directly follows from the proof of the afore mentioned theorem in [44].

Remark 5.3. In the literature the formulation of statements similar to Theorem 5.3 on the DAOE may sometimes have a misleading character. This may occur when the existence of a solution to Eq. (5.9) is stated first, and a subsequent claim is made that any minimising policy in Eq. (5.9) is average cost optimal. Strictly speaking, this may not be true. Examples 12 and 13 below, at the end of this section, illustrate that other 'wrong' solutions may exist. Unfortunately, Assumption 3 does not admit tools to select the 'right' solution among the set of all solutions. Thus, under Assumption 3 a solution to the DAOE should always be obtained via the vanishing discount approach, as in Theorem 5.3 (ii).

The next issue to be discussed is how to verify Assumption 3 (ii) and Assumption 3 (iv). This can be inferred from the following Lyapunov function criterion, which is related to [30, Lemma 3.1].

Lemma 5.2. *Let $\phi \in \Phi$. Suppose that the Markov process with transition matrix P^ϕ has one closed class of states plus possibly transient states. Assume the existence of functions $f, k : S \to (0, \infty)$, a constant K and a finite set $D \subset S$ with*

i) $f(i) \geq \max\{1, c^\phi(i)\}$, $i \in S \setminus D$;
ii) $f(i) + \sum_j p^\phi(j \,|\, i) k(j) \leq k(i) + \mathbf{1}_{\{D\}}(i) \cdot K$, $i \in S$.

Then the closed class is positive recurrent, and in particular there is a constant κ such that

a) $\mu_D^\phi(i), C_D^\phi(i) \leq \kappa \cdot k(i)$, $i \in S$;
b) $\mu_{i_0}^\phi(i), C_{i_0}^\phi(i) < \infty$ *for* $i_0 \in D$ *and* $i \in S$.

For $f \equiv 1$, condition (ii) is simply Foster's criterion for positive recurrence of an irreducible Markov process (cf. [26], if D consists of one state only). The above version is a generalisation designed for MDPs (cf. [30, Lemma 3.1]). Statement (a) is proved by a simple iteration argument, statement (b) is shown by considering the embedded semi-Markov chain on the finite set M, analogously to the proof of [16, Lemma 5.2.3].

Example 10 (Continuation of Examples 1, 2, 4, 5, 8 and 9 Server Allocation) *We consider the non-truncated problem with parameter set $\Phi_{\infty,\infty}$. In order that Theorem 5.3 be applicable, we only need check Assumption 3. Assumptions 3 (i, iii) are clearly satisfied, and we will check (ii, iv) by means of Lemma 5.2.*

Notice, that for any parameter $\phi \in \Phi_{\infty,\infty}$, the associated Markov process has at most one closed class, since the empty state $(0,0)$ is reachable from any other state with positive probability. This means that we have to find a parameter ϕ_0 for which the closed class is positive recurrent, the average expected cost is finite, and the closed class is reached in finite expected time and with finite expected total cost. This then verifies (ii). To verify (iv), it is sufficient that the Markov process associated with ϕ_0 is irreducible.

Under the condition that $\mu_2 < (1-p_1)\mu_1$, *Example 8 (cf. [12, Chap. 6]) evidences that AP1 is* α*-discount optimal for any* $\alpha \in (0,1)$. *The vanishing discount approach then suggests that AP1 must be average cost optimal as well. It therefore seems sensible to fix* ϕ_0 *as the parameter corresponding to the AP1 policy given in (5.8), and check (ii,iv) for this parameter. We do not assume that* $\mu_2 < (1-p_1)\mu_1$. *However, we will see below, that we do have to make (other) assumptions on the parameters in order that the Markov process associated with parameter* ϕ_0 *is positive recurrent.*

Put $k(i) = (1+\psi)^{i_1}(1+\theta)^{i_2}$, *where* $\psi, \theta > 0$ *have to be suitably chosen. Remind, that* $c^{\phi_0}(i) = i_1 + i_2$ *and so* k *has to be increasing in* i, *whence the requirement* $\psi, \theta > 0$. *First we will check that we can choose* $\psi, \theta > 0$

$$P^{\phi_0} k(i) \leq \beta k(i), \quad i \neq (0,0),$$

for some $\beta < 1$. *This is stronger than necessary, but will be convenient later on. By insertion, this implies that we have to verify the existence of* $\beta < 1$ *such that*

$$\begin{aligned} \lambda(1+\psi) + p_1\mu_1 \frac{1+\theta}{1+\psi} + (1-p_1)\mu_1 \frac{1}{1+\psi} + \mu_2 &\leq \beta \\ \lambda(1+\psi) + \mu_1 + \mu_2 \frac{1}{1+\theta} &\leq \beta. \end{aligned} \tag{5.10}$$

Since there are only two different inequalities that should hold, it sufficient to choose $\psi, \theta > 0$ *in such a way that the left-hand side of (5.10) is smaller than 1. Using that* $\lambda + \mu_1 + \mu_2 = 1$, *this reduces to the following inequalities*

$$\frac{\mu_1 - \lambda(1+\psi)}{p_1\mu_1} > \frac{\theta}{\psi} > \frac{\lambda(1+\theta)}{\mu_2}. \tag{5.11}$$

Clearly, for (5.11) to be valid, it is necessary that

$$\mu_1 > \lambda, \quad \frac{\mu_1 - \lambda}{p_1\mu_1} > \frac{\lambda}{\mu_2}. \tag{5.12}$$

That $\mu_1 > \lambda$ *is necessary for positive recurrence, is clear, since otherwise the stochastic process associated with number of products in unit 1 cannot be positive recurrent. Necessity of the second condition for positive recurrence can be verified by a so called 'second vectorfield' analysis (cf. [23]). Under the assumption that (5.12) holds, it is simply verified that* $\psi, \theta > 0$ *can be chosen so that (5.11) and therefore (5.10) applies. Indeed, choose* $\varepsilon_1, \varepsilon_2 > 0$, *such that*

$$\frac{\mu_1 - \lambda(1+\varepsilon_1)}{p_1\mu_1} > \varepsilon_2 > \frac{\lambda(1+\varepsilon_1)}{\mu_2}.$$

Then there exist $\psi, \theta > 0$, $\beta < 1$, *with* $\psi, \theta < \varepsilon_1$ *and* $\theta = \psi\varepsilon_2$, *such that (5.10) holds. Now, put* $f(i) = i_1 + i_2$. *Then* $f(i) + \sum_j p^{\phi_0}(j\,|\,i)k(j) \leq k(i)$, *if*

$$f(i) \le (1-\beta)k(i) = (1-\beta)(1+\psi)^{i_1}(1+\theta)^{i_2}. \tag{5.13}$$

It is straightforwardly checked that there exists a finite set $D \subset S$, such that (5.13) holds true for $i \in S \setminus D$. The existence of a constant K for which condition (ii) of Lemma 5.2 holds, directly follows.

We have arrived at the following conclusion. If (5.12) holds, then Theorem 5.3 applies, and any limit point of α-discounted optimal policies, as $\alpha \to 0$, is average cost optimal. If, additionally, $\mu_2 < (1-p_1)\mu_1$, then AP1 is average cost optimal.

To pave the way for developing a roadmap that allows to obtain structures of average cost optimal policies, we will shortly discuss the applicability of VI. Let us first state the algorithm. Again assume that Φ has the product property.

Algorithm 2 VI for an expected average cost optimal policy

1. Select $V_0 : S \to \mathbf{R}$, set $n = 0$.
2. For each $x \in S$, compute $V_{n+1}(i)$ by

$$V_{n+1} = \min_{\phi \in \Phi} \left\{ c^\phi + P^\phi V_n \right\},$$

and let

$$\phi_{n+1} \in \arg\min_{\phi \in \Phi} \{ c^\phi + P^\phi V_n \}.$$

3. Increment n by 1 and return to step 2.

To our knowledge there are relatively few non-problem specific papers on the convergence of average cost VI for countable state space MDPs, cf. [4, 31, 43], and [2], the latter of which is based on the thesis [46]. The conditions in the first three papers are not restricted to conditions on the input parameters. In our opinion, the easiest verifiable ones are contained in the paper [4], involving properties of the set of policies $\{\phi_n\}_n$. In case of well-structured problems, say ϕ_n are all equal, or have very specific structures, these conditions are easily be verifiable.[1] Here, we will restrict to the conditions from [46] and [2] that are, as far as we know, the only ones formulated directly in terms of the input parameters of the process. The notation $\mathbf{e}$ stands for the function on S identically equal to 1.

Theorem 5.4. *Let Φ be a compact set with the product property. Suppose that the following drift condition, called M-uniform geometric recurrence, holds: there exist a function $M : S \to [1,\infty)$, a finite set $D \subset S$ and a constant $\beta < 1$, such that*

$$\sum_{j \notin D} p^\phi(j \,|\, i) M(j) \le \beta M(i), \quad i \in S, \phi \in \Phi.$$

[1] We have the impression that there is a gap in the proofs in [4].

Suppose further that the following holds as well:

- *Assumption 1;*
- $\sup_{\phi\in\Phi} \|c^\phi\|_M < \infty$*;*
- $\phi \mapsto P^\phi M$ *is component-wise continuous on* Φ*;*
- *the Markov process with transition matrix* P^ϕ *is aperiodic and has one closed class, plus possibly transient states,* $\phi \in \Phi$*.*

Let $0 \in S$ *be a selected state. Then, there is a unique solution pair* (g^*, H^*) *with* $H^* \in \ell^\infty(S,M)$*, and* $H^*(0) = 0$*, to Eq. (5.9) with the properties from Theorem 5.3.*

Furthermore, average cost VI converges, that is, $\lim_{n\to\infty}(V_n - V_n(0)\mathbf{e}) = H^*$*, and any limit point of the sequence* $\{\phi_n\}_n$ *is average cost optimal and minimiser of the DAOE (5.9) with solution tuple* (g^*, H^*)*.*

The V-uniform geometric recurrence condition in Theorem 5.4 has been introduced in [18], and shown in [19] to imply the assertion in Theorem 5.4. The paper [19], see also [46], has derived an equivalence of this condition (under extra continuity conditions) with M-uniform geometric ergodicity. The thesis [46] additionally shows a similar implication for bounded jump Markov decision processes in continuous time, by uniformisation. Both properties have been extensively used both in the case of a parameter space consisting of one element only (cf. [35] and later works), and in the case of product parameter spaces in the context of optimal control. Together with the negative dynamic programming conditions developed by Sennott [42], the M-uniform geometric recurrence and ergodicity, developed in [17] and [18], have become 'standard' conditions in many papers and books. See for instance [27, 28], and [38], as well as references therein, for a survey and two books using both types of conditions. A parametrised version of [19] in both discrete and continuous time is currently in preparation. The drawback of using M-geometric recurrence is its implying each associated Markov process to have only positive recurrent closed classes. This is a major disadvantage for many models and therefore our motivation for using Assumption 3. Note that customer abandonment has a strong stabilising effect on the associated Markov processes, and then M-uniform geometric recurrence typically may apply.

Example 11 (Continuation of Examples 1, 2, 4, 5, 8, 9 and 10 Server Allocation) *As in Example 10, consider the parameter set* $\Phi_{\infty,\infty}$*, that is, the non-truncated server allocation model. It is simply checked that the function k constructed in Example 10 satisfies the uniform geometric recurrence property from Theorem 5.4 for some finite set D and constant* $\beta < 1$*, provided that (5.10) holds. Thus, VI converges for the non-truncated server allocation model.*

5.2.3.1 Roadmap to Structural Properties

Below we formulate a scheme for deriving the structure of an optimal policy and value function, if the optimisation criterion is to minimise the expected average cost. Let $\Phi = \mathcal{A}$ be the (product) set of all stationary, deterministic policies.

Roadmap for average cost MDPs in discrete time

1. Check the conditions of Theorem 5.4.
2. If satisfied then:
   - perform VI Algorithm 2, and check iteratively that the structural properties of interest hold.
3. If not satisfied, then check Assumptions 1, 2(α), for all $\alpha \in (0,1)$, and 3. If satisfied then:
   (a) perform VI Algorithm 1 for the α-discounted cost criterion, and check iteratively that the structural properties of interest hold. If there exists $\alpha_0 > 0$ such that the desired structural properties hold for all $\alpha \in (0, \alpha_0)$ then
   (b) apply the vanishing discount approach by taking the limit $\alpha \to 0$. This is justified by Theorem 5.3. If the limit policy is optimal for all sufficiently small values of α, then it is both average optimal and *strongly Blackwell optimal*.
4. If not satisfied, or if no structural properties can be iteratively concluded, then the outcome is inconclusive.

Note that the vanishing discount approach has the advantage of allowing a conclusion on Blackwell optimality of the limiting policy.

5.2.3.2 Examples Where the DAOE Has No Unique Solution

Here we address the problem that constructing a solution to the DAOE does not guarantee that the solution is related to the optimal value function and optimal average expected cost. This problem does not arise when working in the framework of normed spaces, in the sense that a solution in the space is the 'right' one.

Example 12 *Consider a simple random walk on the state space $S = \mathbf{Z}$ without any control. Thus Φ consists of one element ϕ, say $\phi = 1$. The transition mechanism is given by*

$$p^1(j \mid i) = \begin{cases} \lambda, & j = i+1 \\ \mu, & j = i-1, \end{cases}$$

where $\lambda < \mu$ and $\lambda + \mu = 1$. The cost in state $i \neq 0$ is equal to $c^1(i) = 1$, and $c^1(0) = 0$. This is a transient Markov process, and hence it does not satisfy Assumption 3. However, it does satisfy the assumptions of [42], implying the assertion of Theorem 5.3 to hold.

This implies that the vanishing discount approach yields solution tuple (g,H) *of (5.9), with* $g=1$ *and*

$$H(i)=\begin{cases}\frac{1-(\mu/\lambda)^i}{\mu-\lambda}, & i<0\\ 0, & i\geq 0,\end{cases}$$

if $i_0=0$ *is chosen. This can be deduced by a 'clearing analysis' as in [21].*

However, other solutions $(g=1,H')$ *to (5.9) exist: for any* $\theta\in\Re$

$$H'(i)=\begin{cases}(1-\theta)\frac{1-(\mu/\lambda)^i}{\mu-\lambda}, & i<0\\ 0, & i=0\\ \theta\frac{(\mu/\lambda)^i-1}{\mu-\lambda}, & i>0.\end{cases}$$

There is no a priori tool to determine which solution is the one obtained from the vanishing discount approach.

Example 13 *Next, we restrict the simple random walk to* $S=\mathbf{Z}_+$*, and associate the corresponding transitions with parameter* $\phi^1=1^{\mathbf{Z}_+}$*. In other words,*

$$p^{\phi^1}(j\,|\,i)=\begin{cases}\lambda, & j=i+1\\ \mu, & j=(i-1)^+,\end{cases}$$

where $\lambda<\mu$ *and* $\lambda+\mu=1$*. Suppose that holding cost i is incurred per (discrete) unit time, when the number of customers in the system is i:*

$$c^{\phi^1}(i)=i,\quad i\in S.$$

In [6] it was shown that the equation $v+g=c^{\phi^1}+P^{\phi^1}H$ *has the following solutions: to any* $g\in\Re$ *there is a solution tuple* (g,H_g) *with* $H^g:S\to\Re$ *the function given by*

$$H^g(i)=-\frac{i-1}{\mu-\lambda}g+\frac{(i-1)(i-2)}{2(\mu-\lambda)}+\mu\frac{i-1}{(\mu-\lambda)^2}+\frac{g}{\lambda}+\mu\frac{(\mu/\lambda)^{i-1}-1}{\mu-\lambda}\Big\{\frac{g}{\mu-\lambda}+\frac{g}{\lambda}-\frac{\mu}{(\mu-\lambda)^2}\Big\},\tag{5.14}$$

for $i\geq 1$ *and* $H^g(0)=0$*. The solution obtained from a vanishing discount approach, is the one for which the expression between curly brackets is 0, i.e. for which*

$$\frac{g}{\mu-\lambda}+\frac{g}{\lambda}-\frac{\mu}{(\mu-\lambda)^2}=0,$$

in other words

$$g=\frac{\lambda}{\mu-\lambda},$$

and

$$H^g(i) = -\frac{i-1}{\mu-\lambda}g + \frac{(i-1)(x-2)}{2(\mu-\lambda)} + \mu\frac{i-1}{(\mu-\lambda)^2} + \frac{g}{\lambda}.$$

This can also be derived from boundedness conditions analysed in [7]. Thus, $\mathrm{g}(\phi^1) = \lambda/(\mu-\lambda)$.

Define the following parameter space $\Phi = \{\phi^1, \phi^2\}$*, where* $\phi^2 = \{1,2,1,1,\ldots\}$*. Parameter* ϕ^2 *differs from* ϕ^1 *only in the second coordinate. The corresponding transition probabilities are*

$$p^{\phi^2}(3\,|\,1) = \lambda = 1 - p^{\phi^2}(0\,|\,1),$$

and $p^{\phi^2}(j\,|\,i) = p^{\phi^1}(j\,|\,i)$*, for* $i \neq 1$*. Similarly,* $c^{\phi^2}(i) = c^{\phi^1}(i) = i$*,* $i \neq 1$*. Choose* $c^{\phi^2}(1)$ *small enough (possibly negative) so that* $\mathrm{g}^{\phi^2} < \mathrm{g}^{\phi^1}$.

Then Φ *has indeed the product property and the parametrised Markov process is an MDP. This MDP satisfies Assumption 3. Although the direct cost possibly is not non-negative, it is bounded below.*

We claim that we may choose a constant g so large that

$$\begin{aligned} H^g(1) + g &= c^{\phi^1}(1) + \sum_j p^{\phi^1}(j\,|\,1)H^g(j) \\ &< c^{\phi^2}(1) + \sum_j p^{\phi^2}(j\,|\,1)H^g(j) = c^2(1) + \lambda H^g(3) + \mu H^g(0). \end{aligned}$$

In other words, the minimisation prescribes to choose ϕ^1 *in state 1, when considering the tuple* (g, H_g)*. Indeed, this choice is possible if*

$$c^{\phi^2}(1) + \lambda H^g(3) > 1 + \lambda H^g(2),$$

or

$$1 - c^{\phi^2}(1) < \lambda(H^g(3) - H^g(2)). \tag{5.15}$$

It can be checked that

$$H^g(3) - H^g(2) > \frac{\mu^2}{\lambda^2}\Big(\frac{g}{\lambda} - \frac{\mu}{(\mu-\lambda)^2}\Big).$$

Therefore, one may choose $g > \mathrm{g}^{\phi^1}$ *large enough for Eq. (5.15) to be true. Hence,* (g, H^g) *is a solution to Eq. (5.9) for the MDP with minimising policy* ϕ^1*. However, by construction* $g > \mathrm{g}^{\phi^1} > \mathrm{g}^{\phi^2}$*. Thus,* (g, H^g) *is a solution to the DAOE, where g is not the minimum expected average cost, and the minimising policy is not optimal.*

5.3 Continuous Time Model

In this section we will consider continuous time parametrised Markov processes. The setup is analogous to the discrete time case. Again we consider a parameter space Φ and a countable state space S. With each $\phi \in \Phi$ we associate an $S \times S$ generator matrix or q-matrix Q^ϕ and a cost rate vector $c^\phi : S \to \Re$. Following the construction in [32], see also [36], one can define a measurable space $(\Omega, \mathcal{F})$, a stochastic process $X : \Omega \to \{f : [0,\infty) \to S \,|\, f \text{ right-continuous}\}$, a filtration $\{\mathcal{F}_t\}_t \subset \mathcal{F}$ to which X is adapted, and a probability distribution P_ν^ϕ on $(\Omega, \mathcal{F})$, such that X is the minimal Markov process with q-matrix Q^ϕ, for each initial distribution ν on S and $\phi \in \Phi$.

Denote by $P_t^\phi = \{p_t^\phi(j\,|\,i)\}_{i,j\in S}$, $t \geq 0$, the corresponding minimal transition function and by E_ν^ϕ the expectation operator corresponding to P_ν^ϕ. The elements of Q^ϕ are denoted by $q^\phi(j\,|\,i)$, $i, j \in S$, and $-q^\phi(i\,|\,i)$ is the parameter of the exponentially distributed sojourn time in state i.

Assumption 4

i) Q^ϕ *is a conservative, stable q-matrix, i.e. for* $i \in S$ *and* $\phi \in \Phi$

- $0 \leq q^\phi(i) = -q^\phi(i\,|\,i) < \infty$;
- $\sum_j q^\phi(j\,|\,i) = 0$.

ii) $\{P_t^\phi\}_{t\geq 0}$ *is standard, i.e.* $\lim_{t\downarrow 0} P_t^\phi = \mathbf{I}$, *with* $\mathbf{I}$ *the identity matrix of appropriate dimension.*

iii) $\phi \mapsto q^\phi(j\,|\,i)$ *and* $\phi \mapsto c^\phi(i)$ *are continuous,* $i, j \in S$;

iv) Φ *is locally compact.*

The definition of the product property of $\{Q^\phi\}_\phi$ and $\{c^\phi)\}_\phi$ with respect to Φ is completely analogous to Definition 5.1, and so we omit it. Again, for easy reference, we say that Φ has the product property if $\{Q^\phi\}_\phi$ and $\{c^\phi\}_\phi$ both have the product property with respect to Φ. In this case, the parametrised Markov process is an MDP. Analogously to Remark 5.1, Φ may represent the collection of stationary policies or the stationary, deterministic ones.

Suppose, furthermore, that a lump cost is charged, in addition to a cost rate incurred per unit time. Say at the moment of a jump from i to j lump cost $d^\phi(i,j)$ is charged, when the parameter is ϕ. This can be modelled as a (marginal) cost rate $c^\phi(i) = \sum_{j\neq i} d^\phi(i,j)\, q^\phi(j\,|\,i)$.

Below we give the definitions of various performance measures and optimality criteria. Later on we will provide conditions under which these exist.

For $\alpha > 0$, under parameter $\phi \in \Phi$ the *expected total* α*-discounted cost value function* v^α is given by

$$V_\alpha^\phi(i) = \mathsf{E}_i^\phi\Big[\int_{t=0}^{\infty} e^{-\alpha t} c(X_t)dt\Big], \quad i \in S.$$

Suppose that Φ has the product property. The *minimum expected total* α*-discounted cost* w.r.t Φ is defined as

$$V_\alpha^*(i) = \inf_{\phi \in \Phi} \left\{ V_\alpha^\phi(i) \right\}, \quad i \in S.$$

If $V_\alpha^\phi = V_\alpha^*$, then ϕ is said to be an α-discount optimal policy in Φ.

The *expected average cost* under parameter ϕ is given by

$$\mathrm{g}^\phi(i) = \limsup_{T \to \infty} \frac{1}{T} \mathsf{E}_i^\phi \left[\int_{t=0}^{T} c(X_t) dt \right], \quad i \in S.$$

Suppose that Φ has the product property. The *minimum expected average cost* is defined as

$$\mathrm{g}^*(i) = \inf_{\phi \in \Phi} \left\{ \mathrm{g}^\phi(i) \right\}, \quad i \in S.$$

If $\mathrm{g}^\phi = \mathrm{g}^*$ for some $\phi \in \Phi$ then ϕ is said to be an average cost optimal policy in Φ.

The notions of *Blackwell optimality* and *strong Blackwell optimality* are defined completely analogously to the discrete time versions.

A well-known procedure to determine the structure of an optimal policy in the continuous time case, is to reduce the continuous time MDP to a discrete time MDP in order to be able to apply VI. There are different time-discretisation methods. One is to consider the embedded jump process. Sometimes this is a viable method, see [27] where this approach has been taken. In Sect. 5.3.4 we give an example where the embedded jump approach seems to be less amenable to apply.

Instead, one may use uniformisation. However, applicability hinges on models with bounded jumps are bounded as a function of parameter and state:

$$q := \sup_{i \in S, \phi \in \Phi} q^\phi(i) < \infty. \tag{5.16}$$

This property is violated in models with reneging customers, population models etc, and we will consider how to handle this next.

Let us first recall the uniformisation procedure.

5.3.1 Uniformisation

A detailed account of the uniformisation procedure and proofs can be found in [45]. Uniformisation of time-inhomogeneous Markov processes is studied in [51]. If a continuous time parametrised Markov process has bounded transition rates [cf. Eq. (5.16)], it admits a transformation to an equivalent discrete time parametrised Markov process. Below we list the transformations for the α-discounted and average cost cases.

For the discounted cost criterion the equivalent discrete time process is given by

$$P^\phi = I + \frac{1}{q} Q^\phi, \qquad c^{\phi,d} = \frac{c^\phi}{\alpha + q}, \qquad \alpha^d = \frac{\alpha}{\alpha + q}, \qquad \phi \in \Phi. \tag{5.17}$$

Denote the discrete time α^d-discounted cost under ϕ by as $V_{\alpha^d}^{\phi,d}$. Both the discrete-time and continuous time processes have equal expected discounted cost, i.e.

$$V_{\alpha^d}^{\phi,d} = V_{\alpha}^{\phi}.$$

If Φ has the product property, then this implies that the optimal α- and α^d-discounted value functions with respect to Φ are equal:

$$V_{\alpha^d}^{*,d} = V_{\alpha}^{*}.$$

For the average cost criterion the equivalent discrete time process has transition matrix and immediate cost

$$P^{\phi} = \mathbf{I} + \frac{1}{q}Q^{\phi}, \qquad c^{\phi,d} = \frac{c^{\phi}}{q}, \qquad \phi \in \Phi.$$

Denote the discrete time average cost under parameter ϕ as $\mathrm{g}^{\phi,d}$ and the value function as $H^{\phi,d}$. Under the same parameter, the discrete-time and continuous time expected cost, relate to each other as follows

$$q\mathrm{g}^{\phi,d} = \mathrm{g}^{\phi}.$$

The corresponding value functions are identical:

$$H^{\phi,d} = H^{\phi}.$$

These relations apply similarly to optimal parameters if Φ has the product property.

The main concern is how to proceed in the case of unbounded jump rates $q = \infty$, when the above procedure is not possible.

5.3.2 Discounted Cost

First we treat the discounted cost criterion. This section summarises the results of [13]. The latter paper only treats optimality within the class of stationary Markov policies, as we do in the present chapter. We recall some definitions. These definitions are closely related to the definitions used in the discrete time analysis in Sect. 5.2.1.

Definition 5.4.

- The function $W : S \to (0,\infty)$ is said to be a *moment function*, if there exists an increasing sequence $\{K_n\}_n \subset S$ of finite sets with $\lim_n K_n = S$, such that $\inf_{i \notin K_n} W(i) \to \infty$, as $n \to \infty$.
- Let $\gamma \in \Re$. The function $M : S \to (0,\infty)$ is called a (γ, Φ)-drift function if $Q^{\phi}W \leq \gamma W$ for all $\phi \in \Phi$, where $Q^{\phi}M(i) := \sum_{j \in S} q^{\phi}(j\,|\,i)M(j)$.

Assumption 5 (α)

i) There exist a constant $\gamma < \alpha$ and function $M : S \to (0,\infty)$ such that M is a (γ, Φ)-drift function;
ii) $\sup_\phi \|c^\phi\|_M =: b_M < \infty$ for all $\phi \in \Phi$;
iii) There exist a constant θ and a function $W : S \to (0,\infty)$ such that W is a (θ, Φ)-drift function and W/M is a moment function, where $(W/M)(i) = W(i)/M(i)$, $i \in S$.

Assumptions 5 (α) (i) and 5 (α)(ii) are the continuous time counterpart of Assumption 2 (α). Assumption 5(α)(iii) is sufficient to guarantee nonexplosiveness of the parametrised Markov process (cf. [47, Theorem 2.1]), and implies continuity properties of the map $\phi \mapsto (P_t^\phi M)(i)$, $i \in S$.

Theorem 5.5 ([13, Theorem 4.1]). *Suppose that Assumptions 4 and 5(α) hold, then $\phi \mapsto V_\alpha^\phi$ is component-wise continuous and V_α^ϕ is the unique solution in $\ell^\infty(S,M)$ to*

$$\alpha u = c^\phi + Q^\phi u.$$

Theorem 5.6 ([13, Theorem 4.2]). *Assume that Φ is a compact set with the product property. Suppose further that Assumptions 4, and 5(α) hold. Then V_α^* is the unique solution in $\ell^\infty(S,M)$ to the α-discount optimality equation (CDOE)*

$$\alpha u(i) = \inf_{\phi \in \Phi} \{c^\phi(i) + \sum_j q^\phi(j\,|\,i) u(j)\}, \quad i \in S. \tag{5.18}$$

There exists $\phi^* \in \Phi$ with $\phi^* \in \arg\min_{\phi\in\Phi}\{c^\phi(i) + \sum_j q^\phi(j\,|\,i)u(j)\}$, $i \in S$. Any policy $\phi \in \Phi$ that minimises Eq. (5.18) is optimal in Φ, and it holds that $V_\alpha^\phi = V_\alpha^*$.

As discussed in Sect. 5.2.2, the parameter set may contain a perturbation component. Introducing a perturbation yields a parameter set of the following form $\Phi = \mathcal{N} \times \mathcal{A}$, where N is a perturbation parameter and $\mathcal{A}$ the set of deterministic stationary (or merely stationary) policies.

Corollary 5.2 ([13, cf. Theorem 5.1]). *Let $\Phi = \mathcal{N} \times \mathcal{A}$. Suppose Assumptions 4 and 5(α) hold. Assume that $\{N\} \times \mathcal{A}$ is a compact set (with the product property) for any $N \in \mathcal{N}$. Then the following hold.*

i) $\lim_{N \to N_0} V_\alpha^{*,(N)} = V_\alpha^{*,(N_0)}$.
ii) Any limit point of $(\delta_N^*)_{N \to N_0}$ is optimal in $\{N_0\} \times \mathcal{A}$.
iii) Suppose that the MDP with parameter set $\{N\} \times \mathcal{A}$ is uniformisable, i.e.

$$q^N := \sup_{i \in S, \delta \in \mathcal{A}} |q^{(N,\delta)}(i)| < \infty.$$

Consider the discount discrete-time uniformised MDP, with transition matrices, cost and discount factor given by [cf. Eq. (5.17)]

$$P^{(N,\delta)} = \mathbf{I} + \frac{1}{q^N} Q^{(N,\delta)}, \quad c^{(N,\delta),d} = \frac{c^{(N,\delta)}}{\alpha + q^N}, \quad \alpha^d = \frac{\alpha}{\alpha + q^N}.$$

Then the MDP satisfies Assumptions 1 and 2 (α^d), for the same function M.

Proof. Assertions (i) and (ii) are in fact [13, Theorem 5.1], but they follow easily from Theorems 5.5 and 5.6. Assertion (iii) is a direct verification.

5.3.2.1 Roadmap to Structural Properties

We finally have collected the tools to provide a scheme for the derivation of structural properties of an optimal policy and value function for a continuous time MDP with unbounded jump rates, provided the required conditions hold. Applications of this scheme are discussed in [9] and [14], cf. also [12].

Let $\mathcal{A}$ be the set of stationary, deterministic policies, and $\Phi = \mathcal{N} \times \mathcal{A}$. Each set $\{N\} \times \mathcal{A}$ is assumed to be compact and to have the product property, $N \in \mathcal{N}$.

Roadmap for α-discounted MDPs in continuous time

1. Check Assumptions 4 and 5(α) of Theorem 5.6.
2. If satisfied and $q < \infty$, do
   (a) perform a uniformisation;
   (b) use VI Algorithm 1 to verify the structural properties of an optimal policy and value function;
   (c) use the equivalence of uniformised and non-uniformised systems to obtain the structure of an optimal policy and value function of the non-uniformised continuous time MDP.
3. If satisfied and $q = \infty$, do
   (i) perform a bounded jump perturbation leaving the structural properties intact and satisfying Assumptions 4 and 5(α). For instance, one might apply SRT (see Sect. 5.2.2) or try a brute force perturbation;
   (ii) do steps 2(a, b, c). This potentially yields structural properties of an optimal policy and the value function for each N-perturbed MDP;
   (iii) take the limit for the perturbation parameter to vanish. Corollary 5.2 gives the structural results for an optimal policy and value function.
4. If the assumptions for Theorem 5.6 do not hold, or if no structural properties can be derived, then the outcome is inconclusive.

As has been mentioned already, one might apply discounted VI directly to the associated discrete time MDP, embedded on the jumps of the continuous time MDP (cf. e.g. [27, Theorem 4.12]). In the example of Sect. 5.3.4 we discuss some problems with the application of this procedure.

5.3.3 Average Cost

We finally turn to studying the average cost criterion in continuous time. The assumptions that we make, are Assumption 4 and the analog of Assumption 3 that we used in Sect. 5.2.3 for analysing the average cost criterion in discrete time. In fact, Assumption 3 can be used unaltered. However, one has to use the continuous time definitions of the hitting time of a state, and total expected cost incurred till the hitting time.

The hitting time τ_z of a state $z \in S$ is defined by:

$$\tau_z = \inf_{t>0}\{X_t = z, \exists s \in (0,t) \text{ such that } X_s \neq z\}. \tag{5.19}$$

Then, $m_{iz}(\phi) = \mathsf{E}_i^{\phi}\tau_z$ and $C_z^{\phi}(i) = \mathsf{E}_i^{\phi}\int_0^{\tau_z} c(X_t)dt$, where either expression may be infinite.

The following theorem is completely analogous to the discrete time equivalent Theorem 5.3, with the only difference that the CAOE below has a slightly different form.

Theorem 5.7. *Suppose, that Φ is a compact set with the product property. Furthermore, suppose that Assumptions 4, 5(α), $\alpha > 0$, and 3 hold.*

i) There exists a solution tuple (g^*, H^*) to the average cost optimality equation (CAOE)

$$g = \min_{\phi\in\Phi}\{c^{\phi}(i) + \sum_{j\in S} q^{\phi}(j\,|\,i)u(j)\}, \tag{5.20}$$

with the properties, that

(1) $g^* = \mathrm{g}$ is the minimum expected average cost (in Φ),
(2) $\phi^* \in \Phi$ with

$$\phi^* \in \arg\min_{\phi\in\Phi}\{c^{\phi}(i) + \sum_{j\in S} q^{\phi}(j\,|\,i)u(j)\}$$

is (average cost) optimal in Φ, and
(3) there exists $i^* \in D$ with $H^*(i^*) = \inf_i H^*(i)$.

ii) Let $i_0 \in S$. Any sequence $\{\alpha_n\}_n$ with $\lim_{n\to\infty}\alpha_n = 0$, has a subsequence, again denoted $\{\alpha_n\}_n$, along which the following limits exist:

$$\begin{aligned} H'(i) &= \lim_{n\to\infty}\{V^*_{\alpha_n}(i) - V^*_{\alpha_n}(i_0)\}, \quad i \in S, \\ g' &= \lim_{n\to\infty}\alpha_n V^*_{\alpha_n}(i), \quad i \in S, \\ \phi' &= \lim_{n\to\infty}\phi^{\alpha_n}, \end{aligned}$$

where ϕ^{α_n} is α_n-discount optimal in Φ. Furthermore, the tuple (g', H') is a solution to (5.20) with the properties (1), (2) and (3), so that $g' = \mathrm{g}$. Moreover, ϕ' takes minimising actions in (5.20) for $g = g'$ and $u = H'$.

We have not encountered the above result in this form. However, the derivations are analogous to the discrete time variant, cf. [44, Chap. 7], and to the proofs in [27], where continuous time variants of Sennott's discrete time conditions have been assumed. In fact, Assumption 3 implies [27, Assumption 5.4]. Although one could piece together the proof of Theorem 5.7 from these references, we prefer to give it explicitly in Sect. 5.3.5.

For the verification of Assumption 3 (ii) and Assumption 3 (iv) one may use the following lemma, that is analogous to Lemma 5.2. The proof is similar to the proof of [50, Theorem 1].

Lemma 5.3. *Let $\phi \in \Phi$. Suppose that the Markov process with transition function P_t^ϕ, $t \geq 0$, has one closed class of states. Assume the existence of functions $f, k : S \to [0,\infty)$, a constant K and a finite set $D \subset S$ with*

i) $f(i) \geq \max\{1, c^\phi(i)\}$, $i \in S \setminus \{D\}$;
ii) $f(i) + \sum_j q^\phi(j|i)k(j) \leq 0 + \mathbf{1}_{\{D\}}(i) \cdot K$, $i \in S$.

Then the closed class is positive recurrent and there is a constant κ, such that

a) $\mu_D^\phi(i), C_D^\phi(i) \leq \kappa \cdot k(i)$, $i \in S$;
b) $\mu_{i_0}^\phi(i), C_{i_0}^\phi(i) < \infty$ *for* $i_0 \in D$ *and* $i \in S$.

5.3.4 Roadmap to Structural Properties

First we present a roadmap for determining structural properties of average cost MDPs in continuous time. We illustrate this with a simple example. More complicated examples can be found in [9, 14] and [12].

Then a figure and table will summarise the schematic approach that we have presented through the various roadmaps, including references to the required conditions and results.

Let $\mathcal{A}$ be the set of stationary, deterministic policies, and $\Phi = \mathcal{N} \times \mathcal{A}$. Assume that $\{N\} \times \mathcal{D}$ has the product property for $N \in \mathcal{N}$.

Roadmap for average cost MDPs in continuous time

1. Check Assumptions 4, 5(α), for all $\alpha > 0$, and 3.
2. If satisfied then do
   - apply the roadmap for α-discounted MDPs in continuous time; if the outcome is that the α-discounted problem has the desired structural properties for all $0 < \alpha < \alpha_0$, for some $\alpha_0 > 0$, then do
   - apply the vanishing discount approach of Theorem 5.7 (ii).
3. If the assumptions do not hold, or structural properties can not be shown, the outcome is inconclusive.

Arrival Control of the M/M/1+M-Queue

As an application of this final roadmap, we consider arrival control of the M/M/1+M-queue. Customers arrive in a single server unit with infinite buffer size according to a Poisson (λ) process. Each customer requires an exponentially distributed service time with parameter μ, but he may also renege after an exponentially distributed amount of time with parameter β (service is not exempted from reneging). Arrival process, service times and reneging times are all independent.

Due to reneging, the process associated with the number of customers in the server unit is ergodic at exponential rate. However, having reneging customers is not desirable from a customer service point of view. Therefore, the following arrival control is exercised. Per unit time and per customer a holding cost of size 1 is incurred. The controller can decide to accept (decision A) or reject (decision R) an arriving customer, based on the number of customers present in the system. If he takes decision A, then a lump reward of size K is incurred.

The goal is to select the control policy with minimum expected average cost.

This leads to the following MDP on the state space $S = \mathbf{Z}_+$, where state i corresponds to 0 customers being present in the system. The collection of stationary, deterministic policies is given by $\mathcal{A} = \{\mathsf{A},\mathsf{R}\}^\infty$. We first put $\Phi = \mathcal{A}$ and let $\delta \in \Phi = \mathcal{A}$ be a deterministic stationary policy. The transition rates are as follows: for $i \in S$

$$q^\delta(j\,|\,i) = \begin{cases} \lambda \mathbf{1}_{\{\delta(i)=\mathsf{A}\}}, & j = i+1 \\ \mu + i\beta, & j = i-1, i>0 \\ -(\lambda \mathbf{1}_{\{\delta(i)=\mathsf{A}\}} + \mu \mathbf{1}_{\{i>0\}} + i\beta), & j = i, \end{cases}$$

The lump reward can be modelled as a cost rate, and we get for $i \in S$

$$c^\phi(i) = i - \lambda K \mathbf{1}_{\{\delta(i)=\mathsf{A}\}}.$$

This is an unbounded-rate MDP.

We wish to show that a control-limit acceptance policy is optimal. In other words, that there exists $i^* \in S$, such that accepting in state $i \le i^*$ and rejecting in state $i > i^*$ is average cost optimal.

Denote the always accepting policy by δ_0, then this generates an irreducible Markov process. Let $f(i) = i$, $i \in S$, and $k(i) = e^{\theta i}$, $i \in S$, with $\theta > 0$ and $d \in S$ chosen, such that

$$\lambda e^{2\theta} + i e^{(1-i)\theta} < \mu + i\beta, \quad i \ge d.$$

Then, $f(i) + \sum_j q^{\delta_0}(j\,|\,i)k(j) \le 0$, for $i \ge d$. It easily follows that Lemma 5.3(ii) is satisfied for the finite set $\{0,\ldots,d\}$ and some constant K. Thus $\mathrm{g}^{\delta_0} < \infty$.

Let $\varepsilon > 0$. It then follows that Assumption 3 (i, ii, iii) is satisfied with set $D = \{i\,|\,i - \lambda K \le \mathrm{g}^{\delta_0} + \varepsilon\}$. The verification of Assumption 3 (iv) follows from irreducibility of the Markov process and the renewal-reward theorem.

It is not difficult to verify that Assumptions 4 and 5(α), $\alpha > 0$, are satisfied for $\Phi = \mathcal{A}$. Indeed, for given $\alpha > 0$, there exists $\kappa_\alpha > 0$, such that $M^\alpha(i) = e^{\kappa_\alpha i}$, $i \in S$, is a $(\gamma_\alpha, \mathcal{A})$-drift function for some $\gamma_\alpha > 0$.

It follows that there exists a solution tuple (g^*, H^*) of the CAOE (5.20) with the properties (1), (2), (3). This CAOE takes the form

$$g^* = i + \mathbf{1}_{\{i>0\}}(\mu + i\beta)H^*(i-1) + \lambda \min\{-K + H^*(i+1), H^*(i)\} - (\lambda + \mu \mathbf{1}_{\{i>0\}} + i\beta)H^*(i),$$

where we have already rearranged the terms in such a way that the equation is amenable to analysis. It is easily deduced, that it is optimal to accept in state i if

$$H^*(i+1) - H^*(i) \leq K.$$

Hence, in order that a control-limit acceptance policy be average cost optimal, it is sufficient to show that H^* is convex.

To this end, we will use the roadmap to show that there exists a solution pair (g^*, H^*) to the CAOE (5.20) with properties (1), (2) and (3), and with H^* a convex function. Theorem 5.7 justifies using the vanishing discount approach, and so it is sufficient to show convexity of the α-discount value function V_α, for all $\alpha > 0$ sufficiently small. Note that the imposed conditions for the roadmap for α-discount MDPs are satisfied, since these are imposed as well for the assertions in Theorem 5.7, and these have been checked to hold.

The roadmap for the verification of structural properties of V_α prescribes to choose suitable perturbations. We consider a simple perturbation, where the reneging rates are truncated at $N\beta$ in states $i \geq N$, $N \geq 1$. The value $N = \infty$ then corresponds to the original MDP.

Thus, we consider the extended parameter set $\Phi = \{\mathcal{N}\} \times \mathcal{A}$, where $\mathcal{N} = \{1, 2, \ldots\}$. Put

$$q^{(N,\delta)}(j \mid i) = \begin{cases} \lambda \mathbf{1}_{\{\delta(i)=\mathsf{A}\}}, & j = i+1 \\ \mu + (N \wedge i)\beta, & j = i-1, i > 0 \\ -(\lambda \mathbf{1}_{\{\delta(i)=\mathsf{A}\}} + \mathbf{1}_{\{i>0\}}(\mu + (N \wedge i)\beta), & j = i, \end{cases}$$

for $(N, \delta) \in \Phi$. Then, M_α is a (γ_α, Φ)-drift function, and Assumptions 4 and 5(α) are satisfied, $\alpha > 0$, for this extended parameter space Φ.

Fix $\alpha > 0$ and $N \in \{1, 2, \ldots\}$. By virtue of Corollary 5.2 it is sufficient to check convexity of the α-discount value function $V_\alpha^{(N)}$, for the N-perturbation. Finally, by Theorem 5.2 it is sufficient to check convexity of $V_{\alpha,n}(N)$, which is the n-horizon approximation of $V_\alpha^{(N)}$.

Convexity of $V_{\alpha,n}^{(N)}$ follows iteratively by putting $V_{\alpha,0}^{(N)} \equiv 0$, and checking that convexity is propagated through the iteration step: for $i \in S$

$$\begin{aligned} V_{\alpha,n+1}^{(N)}(i) = & i - \alpha(\mu \mathbf{1}_{\{i>0\}} + (i \wedge N)\beta)V_{\alpha,n}^{(N)}(i-1) \\ & + \alpha\lambda \min\{-K + V_{\alpha,n}^{(N)}(i+1), V_{\alpha,n}^{(N)}(i)\} \\ & + \alpha(1 - \lambda - \mu \mathbf{1}_{\{i>0\}} + (i \wedge N)\beta)V_{\alpha,n}^{(N)}(i). \end{aligned} \tag{5.21}$$

Event based dynamic programming (cf. [33] and [12, Chap. 7]) applied to Eq. (5.21) yields precisely the propagation of convexity.

Associated Embedded Jump MDP

Instead of introducing a perturbation, we could have applied discounted VI to the associated α-discounted embedded jump MDP. The assumptions that we have made, imply convergence to the α-discounted value function (cf. [27, Theorem 4.14]). This yields the following VI-scheme:

$$\begin{aligned}\bar{V}_{\alpha,n+1}(i) = &\frac{1}{\alpha+\mu\mathbf{1}_{\{i>0\}}+i\beta}\min\Big\{\big(i+(\mu\mathbf{1}_{\{x>0\}}+i\beta)\bar{V}_{\alpha,n}(i-1)\big),\\ &\big(i-\lambda K+\lambda\bar{V}_{\alpha,n}(i+1)+(\mu\mathbf{1}_{\{i>0\}}+i\beta)\bar{V}_{\alpha,n}(i-1)\big)\Big\}.\end{aligned}$$

First note that starting the iterations with the simple function $\bar{V}_{\alpha,0}\equiv 0$, yields a concave function

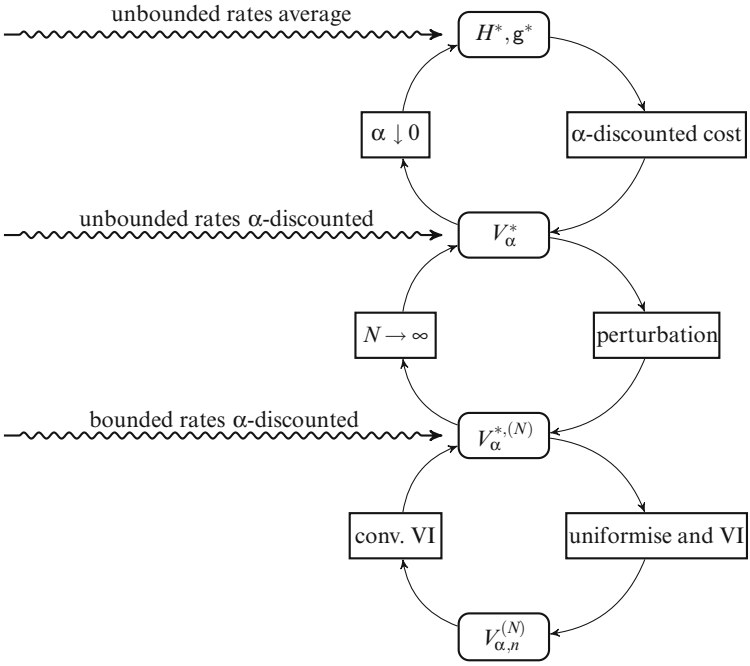
$$\bar{V}_{\alpha,1}(i)=\frac{i-\lambda K}{\alpha+\lambda+\mu\mathbf{1}_{\{i>0\}}+i\beta},\quad i=0,1,\ldots,$$

unless $\beta=0$. A clever choice of $\bar{V}_{\alpha,0}$ could remedy this. However, there will always be the problem that the three functions values that have to be compared in order to propagate convexity, have different denominators.

Note that applying VI on the average cost embedded jump MDP has the same disadvantages. Additionally, one needs extra conditions (cf. Theorem 5.4) to ensure that average VI converges at all.

5.3.4.1 Roadmap Summary

The next figure and table summarise the different roadmaps, with the appropriate references to the results justifying the various steps. In the figure, the limit as the perturbation parameter vanishes is represented by $N\to\infty$.



Summarising table

Time	Criterion	Roadmap			
DT	disc.	VI1			
		Theorem 5.2			
DT	average	VI2			
Vgeo		Theorem 5.4			
DT	average	VDA	then VI1		
no Vgeo		Theorem 5.3	Theorem 5.2		
CT	disc.	UNI	then VI1		
bdd.		Sect. 5.3.1	Theorem 5.2		
CT	disc.	PB	then UNI	then VI1	
unb.		Corollary 5.2	Sect. 5.3.1	Theorem 5.2	
CT	average	VDA	then UNI	then VI1	
bdd.		Theorem 5.7	Sect. 5.3.1	Theorem 5.2	
CT	average	VDA	then PB	then UNI	then VI1
unb.		Theorem 5.7	Corollary 5.2	Sect. 5.3.1	Theorem 5.2

Abbreviations in the summarising table	
Discrete time	DT
Continuous time	CT
Bounded or unbounded rates	bdd. or unb.
α-discounted	disc.
Value iteration algorithm 1 or 2	VI1 or VI2
Vanishing discount approach	VDA
Uniformisation	UNI
Perturbation	PB
Conditions Theorem 5.4	Vgeo

5.3.5 *Proofs*

For the proof of Theorem 5.7 we will need a number of preparatory lemmas. In this section we assume that Φ is a compact set with the product property.

Lemma 5.4. *Suppose that Assumptions 4, 5(α), $\alpha > 0$, and 3 hold. The following hold.*

i) Let μ^{ϕ_0} denote the stationary distribution under parameter ϕ_0, where ϕ_0 has been specified in Assumption 3. Then ϕ_0 has one closed class, R say, that is positive recurrent. It holds, that

$$\mathrm{g}^{\phi_0} = \alpha \sum_R \mu^{\phi_0}(i) V_\alpha^{\phi_0}(i).$$

ii) Let $\phi \in \Phi$. Let $x \notin D$, and put $\tau := \tau_D$ to be the hitting time of D (cf. Eq. (5.19)). Then

$$V_\alpha^{\phi}(i) \geq \mathsf{E}_i^{\phi}\left[\mathbb{1}_{\{\tau=\infty\}} \frac{\mathrm{g}(\phi_0)+\varepsilon}{\alpha} + \mathbb{1}_{\{\tau<\infty\}}\left((1-e^{-\alpha\tau})\frac{\mathrm{g}^{\phi_0}+\varepsilon}{\alpha} + e^{-\alpha\tau} V_\alpha^{\phi}(X_\tau)\right)\right]. \tag{5.22}$$

iii) There exists $i_\alpha \in D$ with $V_\alpha^*(i_\alpha) = \inf_i V_\alpha^*(i)$.

Proof. First we prove (i). By virtue of Assumption 3 (ii) the Markov process associated with ϕ_0 has one closed class, which is positive recurrent. Furthermore, absorption into this class takes place in finite expected time and with finite expected cost, for any initial state $i \notin R$, since necessarily $i_0 \in R$.

Then we get

$$\begin{aligned}
\sum_{i\in R} \mu^{\phi_0}(i) \mathsf{E}_i^{\phi_0}\left[c(X_t)\right] &= \sum_{i\in R} \mu^{\phi_0}(i) \sum_{j\in R} p_t^{\phi_0}(j\,|\,i) c^{\phi_0}(j) \\
&= \sum_{j\in R} c^{\phi_0}(j) \sum_{i\in R} \mu^{\phi_0}(i) p_t^{\phi_0}(j\,|\,i) \\
&= \sum_{j\in R} c^{\phi_0}(j) \mu^{\phi_0}(j) = \mathrm{g}^{\phi_0},
\end{aligned}$$

where the interchange of summation is allowed by nonnegativity. This is used as well to justify the next derivation

$$\begin{aligned}\alpha \sum_{i\in R} \mu^{\phi_0}(i) V_\alpha^{\phi_0}(i) &= \alpha \sum_{i\in R} \mu^{\phi_0}(i) \mathsf{E}_i^{\phi_0}\Big[\int_{t=0}^{\infty} e^{-\alpha t} c(X_t)dt\Big] \\ &= \alpha \int_{t=0}^{\infty} e^{-\alpha t} \sum_{i\in R} \mu^{\phi_0}(i) \mathsf{E}_i^{\phi_0}\big[c(X_t)\big]dt \\ &= \alpha \int_{t=0}^{\infty} e^{-\alpha t} g^{\phi_0} dt = \mathrm{g}^{\phi_0}.\end{aligned}$$

The proof of (ii) follows by splitting the α-discounted cost into three terms, the first two of which represent the α-discounted cost till τ, in the respective cases $\tau = \infty$ and $\tau < \infty$, and the third is the cost starting from $\tau < \infty$:

$$\begin{aligned}V_\alpha^{\phi}(i) =& \mathsf{E}_i^{\phi}\Big[\int_{t=0}^{\infty} e^{-\alpha t} c(X_t)dt\Big] \\ \geq& \mathsf{E}_i^{\phi}\Big[\mathbb{1}_{\{\tau=\infty\}} \int_{t=0}^{\infty} e^{-\alpha t} dt(\mathrm{g}^{\phi_0}+\varepsilon) \\ &\quad + \mathbb{1}_{\{\tau<\infty\}}\Big(\int_{t=0}^{\tau} e^{-\alpha t} dt(\mathrm{g}^{\phi_0}+\varepsilon) + \int_{t=\tau}^{\infty} e^{-\alpha t} c(X_t)dt\Big)\Big] \\ =& \mathsf{E}_i^{\phi}\Big[\mathbb{1}_{\{\tau=\infty\}} \frac{\mathrm{g}^{\phi_0}+\varepsilon}{\alpha} + \mathbb{1}_{\{\tau<\infty\}}\big((1-e^{-\alpha\tau})\frac{\mathrm{g}^{\phi_0}+\varepsilon}{\alpha} + e^{-\alpha\tau} V_\alpha^{\phi}(X_\tau)\big)\Big].\end{aligned}$$

The inequality is due to the definitions of D and τ.

We finally prove (iii). Part (i) implies the existence of $z_\alpha \in R$ such that $\mathrm{g}^{\phi_0} \geq \alpha V_\alpha^{\phi_0}(z_\alpha)$. Then there also exists a $i_\alpha \in D$ with $\mathrm{g}^{\phi_0} \geq \alpha V_\alpha^{\phi_0}(i_\alpha)$. Indeed, suppose such $i_\alpha \in D$ does not exist. Then $V_\alpha^{\phi_0} > \frac{\mathrm{g}^{\phi_0}}{\alpha}$ for all $j \in D$. This leads to a contradiction, since by virtue of part (ii)

$$\frac{\mathrm{g}^{\phi_0}}{\alpha} \geq V_\alpha^{\phi_0}(z_\alpha) \geq \mathsf{E}_{z_\alpha}^{\phi}\Big[(1-e^{-\alpha\tau})\frac{\mathrm{g}^{\phi_0}+\varepsilon}{\alpha} + e^{-\alpha\tau} V_\alpha^{\phi_0}(X_\tau)\Big] > \frac{\mathrm{g}^{\phi_0}}{\alpha}.$$

Let $i_\alpha = \arg\min_{j\in D} V_\alpha^*(j)$, and so $V_\alpha^*(i_\alpha) \leq V_\alpha^{\phi_0}(i_\alpha) \leq \frac{g^{\phi_0}}{\alpha}$. Then $i_\alpha = \arg\min_j V_\alpha^*(j)$, because by Eq. (5.22) for any $i \notin D$ and α-discount optimal policy ϕ_α

$$\begin{aligned}V_\alpha^*(i) &= V_\alpha^{\phi_\alpha}(i) \\ &\geq \mathsf{E}_i^{\phi_\alpha}\Big[\mathbb{1}_{\{\tau=\infty\}} \frac{\mathrm{g}^{\phi_0}+\varepsilon}{\alpha} + \mathbb{1}_{\{\tau<\infty\}}\big((1-e^{-\alpha\tau})\frac{\mathrm{g}^{\phi_0}+\varepsilon}{\alpha} + e^{-\alpha\tau} V_\alpha^*(X_\tau)\big)\Big] \\ &\geq \mathsf{E}_i^{\phi}\Big[\mathbb{1}_{\{\tau=\infty\}} V_\alpha^*(i_\alpha) + \mathbb{1}_{\{\tau<\infty\}}\big((1-e^{-\alpha\tau})V_\alpha^*(i_\alpha) + e^{-\alpha\tau} V_\alpha^*(i_\alpha)\big)\Big] \\ &= V_\alpha^*(i_\alpha).\end{aligned}$$

Lemma 5.5. *Suppose that Assumptions 4, 5 (α), $\alpha > 0$, and 3 hold. Let $\{\alpha_n\}_n$ be a positive sequence converging to 0. The following hold.*

i) There exist a subsequence, call it $\{\alpha_n\}_n$ *again, and* $i_0 \in D$ *such that* $\alpha_n V^*_{\alpha_n}(i_0) \le \mathrm{g}^{\phi_0}$, $n = 1, 2, \ldots$.

ii) There exist a constant L *and a function* $U : S \to (0, \infty)$, *such that* $-L \le V^*_\alpha(i) - V^*_\alpha(z) \le U(i)$, $\alpha > 0$, *where* z *is as in Assumption 3(ii).*

Proof. To prove (i), note that Lemma 5.4 (iii) implies for all n the existence of $i_{\alpha_n} \in D$, such that $V^*_{\alpha_n}(i_{\alpha_n}) \le V^*_\alpha(i)$, $i \in S$. By Assumption 3 (iii) D is finite, and so there exists $i_0 \in D$ and a subsequence of $\{\alpha_n\}_n$, that we may call $\{\alpha_n\}_n$ again, such that $i_{\alpha_n} = i_0$. Therefore by Lemma 5.4 (i), for all n

$$\alpha_n V^*_{\alpha_n}(i_0) \le \alpha_n \sum_i \mu^{\phi_0}(i) V^*_{\alpha_n}(i) \le \alpha_n \sum_i \mu^{\phi_0}(i) V^*_\alpha(i) = \mathrm{g}^{\phi_0}.$$

For the proof of (ii), take

$$U(i) = C^{\phi_0}_z(i), \quad L = \max_{j \in D} C^{\phi^j}_j(z),$$

with ϕ^j from Assumptions 3 (iv). Let $\alpha > 0$. Let strategy ϕ follow ϕ_0 until z is reached, from then onwards it follows the α-discount optimal policy ϕ_α. Then again by Assumption 3 (ii) we have

$$V^*_\alpha(i) \le V^\phi_\alpha(i) \le C^{\phi_0}_z(i) + V^{\phi_\alpha}_\alpha(z) = C^{\phi_0}_z(i) + V^*_\alpha(z).$$

Notice that Assumptions 3 (iii) and (iv) yield $L < \infty$. According to Lemma 5.4 (iv) there is a minimum cost starting state $i_\alpha \in D$. Let ϕ' be the policy that uses policy ϕ^{i_α} of Assumption 3 (iv) until hitting i_α, after which ϕ' follows the α-discount optimal policy ϕ_α. This yields,

$$V^*_\alpha(z) - V^*_\alpha(i) \le V^*_\alpha(z) - \min_i V^*_\alpha(i) \le V^\phi_\alpha(z) - V^*_\alpha(i_\alpha) \le C^{\phi^{i_\alpha}}_z(i_\alpha) \le L.$$

Lemma 5.6. *Suppose that Assumptions 4, 5*(α), $\alpha > 0$, *and 3 hold. Then,*

$$\limsup_{\alpha \downarrow 0} \alpha V^*_\alpha(i) \le \mathrm{g}^\phi(i), \quad i \in S, \phi \in \Phi.$$

Proof. Let $\phi \in \Phi$. We wish to apply Theorem 5.8 for $s(t) = \sum_j p^\phi_t(j \,|\, i) c^\phi(j)$. First, Assumption 4, Assumption 5(α) and the dominated convergence theorem yield that $t \mapsto \sum_j p^\phi_t(j \,|\, i) c^\phi(j)$ is continuous and $|V^\phi_\alpha(i)| < \infty$ (cf. Theorem 5.5). By Assumption 3 (i),

$$\sum_j p^\phi_t(j \,|\, i) c^\phi(j),\ V^\phi_\alpha(i) \ge 0, \quad i \in S.$$

Then, $S(\alpha) = V^\phi_\alpha(i)$ and $\mathrm{g}^\phi(i) = \limsup_{T \to \infty} \frac{1}{T} S_T$. Hence Theorem 5.8 (1c) implies

$$\limsup_{\alpha \downarrow 0} \alpha V^\phi_\alpha(i) \le \mathrm{g}^\phi(i).$$

Lemma 5.7 ([27, Theorem 5.2]). *Suppose that Assumptions 4, 5(α), $\alpha > 0$, and 3 hold. Let (g,H) be a tuple, with $g \in \Re$ and $H : S \to [-L,\infty)$, $i \in S$, and $\phi \in \Phi$ be such that*

$$g \geq c^{\phi}(i) + \sum_j q^{\phi}(j\,|\,i)H(j), \quad i \in S.$$

Then $\mathrm{g}^{\phi}(i) \leq g$, $i \in S$.

Proof. The proof is identical to the proof of [27, Theorem 5.2].

Now we have all results at hand to finish the proof of Theorem 5.7. The most important difficulty is to obtain the CAOE from a continuous time average cost optimality inequality (CAOI). To achieve this we have translated a very interesting argument used in [44, Chap. 7] for the discrete time case to continuous time.

Proof of Theorem 5.7. Let $\{\alpha_n\}_n > 0$ be a positive sequence converging to 0. Lemma 5.5(ii) implies that $-L \leq V^*_{\alpha_n}(i) - V^*_{\alpha_n}(z) \leq U(i)$, for a constant L and a function $U : S \to (0,\infty)$, and for $i \in S$. Note that $[-L,U(i)]$ is compact. By a diagonalisation argument, the sequence has a convergent subsequence, denoted $\{\alpha_n\}_n$ again, along which the limit exists for any $i \in S$, say $V^*_{\alpha_n}(i) - V^*_{\alpha_n}(z) \to H'(i)$, $i \in S$.

Lemma 5.5(i) implies that there exists a further subsequence, again denoted $\{\alpha_n\}_n$, such that $0 \leq \alpha_n V^*_{\alpha_n}(i_0) \leq \mathrm{g}^{\phi}_0$, for some $i_0 \in D$. Compactness of $[0,\mathrm{g}^{\phi}_0)]$ implies existence of a limit point, say g', along a subsequence, that in turn is denoted by $\{\alpha_n\}_n$.

By the above, $\lim_{n\to\infty} \alpha_n(V^*_{\alpha_n}(j) - V^*_{\alpha_n}(i_0) = 0$, and thus $\alpha_n V^*_{\alpha_n}(j) \to g'$ for all $j \in S$.

Since Φ is compact, there is a final subsequence of $\{\alpha_n\}_n$, denoted likewise, such that $\{\phi^{\alpha_n}\}_n$, with ϕ^{α_n} α-discount optimal, has a limit point ϕ' say. The tuple (g',H') has property (3) from part (i) of the Theorem by Lemma 5.4.

We will next show that this tuple is a solution to the following inequality:

$$g' \geq c^{\phi'}(i) + \sum_j q^{\phi'}(j\,|\,i)H'(j) \geq \inf_{\phi\in\Phi}\{c^{\phi}(i) + \sum_j q^{\phi}(j\,|\,i)H'(j)\}, \quad i \in S. \quad (5.23)$$

Indeed, the α-DDOE (5.18) yields for all $i \in S$

$$\alpha V^*_{\alpha}(i) = c^{\phi_\alpha}(i) + \sum_j q^{\phi_\alpha}(j\,|\,i)V^*_{\alpha}(j).$$

Then we use Fatou's lemma and obtain

$$\begin{aligned}
g' &= \liminf_{n\to\infty}\{\alpha_n V^*_{\alpha_n}(i)\} \\
&= \liminf_{n\to\infty}\Big\{c^{\phi_{\alpha_n}}(i) + \sum_{j\neq i} q^{\phi_{\alpha_n}}(j\,|\,i)[V^*_{\alpha_n}(j) - V^*_{\alpha_n}(z)] - q^{\phi_{\alpha_n}}(i)[V^*_{\alpha_n}(i) - V^*_{\alpha_n}(z)]\Big\} \\
&\geq c^{\phi'}(i) + \sum_{j\neq i} \liminf_{n\to\infty}\{\, q^{\phi_{\alpha_n}}(j\,|\,i)[V^*_{\alpha_n}(j) - V^*_{\alpha_n}(z)]\} \\
&\qquad\qquad - \liminf_{n\to\infty}\big\{q^{\phi_{\alpha_n}}(i)[V^*_{\alpha_n}(i) - V^*_{\alpha_n}(z)]\big\}
\end{aligned}$$

$$
\begin{aligned}
&= c^{\phi'}(i) + \sum_j q^{\phi'}(j\,|\,i) H'(j) \\
&\geq \inf_{\phi \in \Phi} \{ c^{\phi}(i) + \sum_j q^{\phi}(j\,|\,i) H'(j) \},
\end{aligned}
$$

where subtraction of $V^*_{\alpha_n}(z)$ is allowed, since $Q(\phi)$ has row sums equal to zero. In the third equation we use continuity of $\phi \mapsto c^{\phi}(i)$ and $\phi \mapsto q^{\phi}(j\,|\,i)$.

This allows to show that (g', H') has property (1) from the Theorem and that ϕ' is optimal in Φ'. Indeed, Lemmas 5.6 and 5.7 yield for all $i \in S$

$$
\mathrm{g}^{\phi'} \leq g' = \lim_{n \to \infty} \alpha_n V^*_{\alpha_n}(i) \leq \mathrm{g}^*(i) \leq \mathrm{g}^{\phi'}(i). \tag{5.24}
$$

Hence $\mathrm{g}^{\phi'}(i) = \mathrm{g}^*(i) = g'$, $i \in S$, and so ϕ' is optimal in Φ, and g' is the minimum expected average cost.

The following step is to show that both inequalities in Eq. (5.23) are in fact equalities. To this end, it is sufficient to show that (g', H') is a solution tuple to the CAOE (5.20). Then Eq. (5.23) immediately implies that ϕ' takes minimising actions in Eq. (5.20) for the solution (g', H').

Hence, let us assume the contrary. If $g' > \inf_{\phi \in \Phi} \{ c^{\phi}(i) + \sum_j q^{\phi}(j\,|\,i) H'(j) \}$ then there exists $\bar{\phi} \in \Phi$, such that $g' > c^{\bar{\phi}}(i) + \sum_j q^{\bar{\phi}}(j\,|\,i) H'(j)$ for at least one state $i' \in S$, say. Put $d(i) \geq 0$ to be the corresponding discrepancy, i.e.

$$
g' = c^{\bar{\phi}} + d(i) + \sum_j q^{\bar{\phi}}(j\,|\,i) H'(j), \quad i \in S. \tag{5.25}
$$

In other words,

$$
0 = c^{\bar{\phi}}(i) + d(i) - g' + \sum_j q^{\bar{\phi}}(j\,|\,i) H'(j), \quad i \in S.
$$

For $i \notin D$, $c^{\bar{\phi}}(i) + d(i) - g' \geq \mathrm{g}^{\phi_0} + \varepsilon - g' \geq \varepsilon$, and so $H' + L\mathbf{e}$ is a non-negative solution to the equation

$$
\sum_j q^{\bar{\phi}}(j\,|\,i)(H'(j) + L) \leq -\varepsilon, \quad j \notin D.
$$

This is precisely the condition in [50, Theorem 1] with $\lambda = 0$.[2] Following the proof of that theorem and using that $q^{\bar{\phi}}(i) > 0$ for $i \notin D$ (otherwise $\mathrm{g}^{\bar{\phi}}(i) = c^{\bar{\phi}}(i) > g'$), we can conclude that

$$
H'(i) + L \geq m^{\bar{\phi}}_D(i), \quad i \notin D,
$$

so that $m^{\bar{\phi}}_D(i) < \infty$, for $i \notin D$.

[2] The factor λ in front of y_i in that paper has been mistakenly omitted.

For $i \in D$, either $q^{\bar{\phi}}(i) = 0$, or $q^{\bar{\phi}}(i) > 0$ and

$$m_D^{\bar{\phi}}(i) = \frac{1}{q^{\bar{\phi}}(i)} + \sum_{j \notin D} \frac{q^{\bar{\phi}}(j\,|\,i)}{q^{\bar{\phi}}(i)} m_D^{\bar{\phi}}(j) \leq \frac{1}{q^{\bar{\phi}}(i)} + \sum_{j \notin D} \frac{q^{\bar{\phi}}(j\,|\,i)}{q^{\bar{\phi}}(i)} (H'(j) + L) < \infty,$$

by virtue of Eq. (5.25). We now will perform an iteration argument along the same lines as the proof of [50, Theorem 1] to show that $d(i) = 0$ for all $i \in S$.

For $i \in D$ with $q^{\bar{\phi}}(i) = 0$, necessarily $g' = c^{\bar{\phi}}(i)$ and then equality in Eq. (5.23) immediately follows.

Thus, it is sufficient to consider the case that $q^{\bar{\phi}}(i) > 0$. Dividing Eq. (5.25) for state i by $q^{\bar{\phi}}(i)$ we get, after reordering,

$$H'(i) \geq \frac{c^{\bar{\phi}}(i) + d(i) - g'}{q^{\bar{\phi}}(i)} + \sum_{j \neq i} \frac{q^{\bar{\phi}}(j\,|\,i)}{q^{\bar{\phi}}(i)} H'(j).$$

Introduce the substochastic matrix P on $S \setminus D$ by

$$p(j\,|\,i) = \begin{cases} \dfrac{q^{\bar{\phi}}(j\,|\,i)}{q^{\bar{\phi}}(i)} & j \notin D \cup \{i\} \\ 0 & \text{otherwise.} \end{cases}$$

This is the taboo jump transition matrix associated with $\bar{\phi}$, with taboo set D. Denote the n iterate by $P^{(n)}$, where $P^{(0)}$ is the $S \times S$ identity matrix. Then, for $i \notin D$ we get

$$\begin{aligned} H'(i) \geq & \frac{c^{\bar{\phi}}(i) + d(i) - g'}{q^{\bar{\phi}}(i)} + \sum_j p(j\,|\,i) H'(j) + \sum_{j \in D} \frac{q^{\bar{\phi}}(j\,|\,i)}{q^{\bar{\phi}}(i)} H'(j) \\ \geq & \frac{c^{\bar{\phi}}(i) + d(i) - g'}{q^{\bar{\phi}}(i)} + \sum_{j \in D} \frac{q^{\bar{\phi}}(j\,|\,i)}{q^{\bar{\phi}}(i)} H'(j) \\ & + \sum_j p(j\,|\,i) \Big[\frac{c^{\bar{\phi}}(j) + d(j) - g'}{q^{\bar{\phi}}(j)} + \sum_w p(w\,|\,j) H'(w) + \sum_{w \in D} \frac{q^{\bar{\phi}}(w\,|\,j)}{q^{\bar{\phi}}(j)} H'(w) \Big] \\ \geq & \sum_{n=0}^{N-1} \sum_j p^{(n)}(j\,|\,i) \frac{c^{\bar{\phi}}(j) + d(j) - g'}{q^{\bar{\phi}}(j)} + \sum_{n=0}^{N-1} \sum_j p^{(n)}(j\,|\,i) \sum_{w \in D} \frac{q^{\bar{\phi}}(w\,|\,j)}{q^{\bar{\phi}}(j)} H'(w) \\ & + \sum_j p^{(N)}(j\,|\,i) H'(j). \end{aligned}$$

Taking the liminf $N \to \infty$, we get

$$\begin{aligned} H'(i) \geq \sum_{n=0}^{\infty} \sum_j p^{(n)}(j\,|\,i) \frac{c^{\bar{\phi}}(j) + d(j) - g'}{q^{\bar{\phi}}(j)} + \sum_{n=0}^{\infty} \sum_j p^{(n)}(j\,|\,i) \sum_{w \in D} \frac{q^{\bar{\phi}}(w\,|\,j)}{q^{\bar{\phi}}(j)} H'(w) \\ + \liminf_{N \to \infty} \sum_j p^{(N)}(j\,|\,i) H'(j). \end{aligned}$$

Clearly

$$\liminf_{N\to\infty}\sum_j p^{(N)}(j\,|\,i)H'(j) \geq \liminf_{N\to\infty}\sum_j p^{(N)}(j\,|\,i)(-L).$$

However, since $m_D^{\bar\phi}(i) < \infty$, $i \notin D$, we get that $\liminf_{N\to\infty}\sum_j p^{(N)}(j\,|\,i) = 0$. Hence, for $\tau := \tau_D$

$$\begin{aligned} H'(i) &\geq \sum_{n=0}^{\infty}\sum_j p^{(n)}(j\,|\,i)\frac{c^{\bar\phi}(j)+d(j)-g'}{q^{\bar\phi}(j)} + \sum_{n=0}^{\infty}\sum_j p^{(n)}(j\,|\,i)\sum_{w\in D}\frac{q^{\bar\phi}(w\,|\,j)}{q^{\bar\phi}(j)}H'(w) \\ &\geq \mathsf{E}_i^{\bar\phi}\Big[\int_{t=0}^{\tau}(c(X_t)+d(X_t)-g')dt\Big] + \mathsf{E}_i^{\bar\phi}\Big[H'(X_\tau)\Big] \\ &= c_D^{\bar\phi}(i) + \mathsf{E}_i^{\bar\phi}\Big[\int_{t=0}^{\tau} d(X_t)dt\Big] - m_D^{\bar\phi}(i)g' + \mathsf{E}_i^{\bar\phi}\Big[H'(X_\tau)\Big], \end{aligned} \tag{5.26}$$

for $i \notin D$. For $i \in D$ with $q^{\bar\phi}(i) > 0$, we can derive the same inequality.

On the other hand, we have that

$$V_\alpha(i) \leq c_D^{\bar\phi}(i) + \mathsf{E}_i^{\bar\phi}\big[e^{-\alpha\tau}V_\alpha^*(X_\tau)\big].$$

This is equivalent to

$$V_\alpha^*(i) - V_\alpha^*(z) \leq c_D^{\bar\phi}(i) - V_\alpha^*(z)(1-\mathsf{E}_i^{\bar\phi}\big[e^{-\alpha\tau}\big]) + \mathsf{E}_i^{\bar\phi}\big[e^{-\alpha\tau}(V_\alpha^*(X_\tau)-V_\alpha^*(z))\big].$$

Hence, for the sequence $\{\alpha_n\}_n$ we have

$$V_{\alpha_n}^*(i) - V_{\alpha_n}^*(z) \leq c_D^{\bar\phi}(i) - \alpha_n V_{\alpha_n}^*(z)\frac{1-\mathsf{E}_i^{\bar\phi}\big[e^{-\alpha_n\tau}\big]}{\alpha_n} + \mathsf{E}_i^{\bar\phi}\big[e^{-\alpha_n\tau}(V_{\alpha_n}^*(X_\tau)-V_{\alpha_n}^*(z))\big].$$

Taking the limit of n to infinity yields

$$\begin{aligned} H'(i) &\leq c_D^{\bar\phi}(i) - g'\cdot m_D^{\bar\phi}(i) + \lim_{n\to\infty}\big\{\mathsf{E}_i^{\bar\phi}\big[e^{-\alpha_n\tau}(V_{\alpha_n}^*(X_\tau)-V_{\alpha_n}^*(z)\big]\big\} \\ &= c_D^{\bar\phi}(i) - g'\cdot m_D^{\bar\phi}(i) + \mathsf{E}_i^{\bar\phi}\big[H'(X_\tau)\big]. \end{aligned} \tag{5.27}$$

Taking the limit through the expectation is justified by the dominated convergence theorem, since

$$|\mathsf{E}_i^{\bar\phi}\big[e^{-\alpha_n\tau}(V_{\alpha_n}^*(X_\tau)-V_{\alpha_n}^*(z))\big]| \leq \mathsf{E}_i^{\bar\phi}e^{-\alpha_n\tau}|V_{\alpha_n}^*(X_\tau)-V_{\alpha_n}^*(z)| \leq \mathsf{E}_i^{\bar\phi}(U(X_\tau)\vee L) < \infty.$$

Combining Eqs. (5.26) and (5.27) yields $d \equiv 0$, for i with $q^{\bar\phi}(i) > 0$.

Since there is equality for ϕ', this also implies that the inf is a min, and so we have obtained that (g', H') is a solution to the CAOE (5.20).

The only thing left to prove, is that the solution tuple (g', H') has property (2), that is, every minimising policy in Eq. (5.20) is average cost optimal. But this follows in the same manner as the argument leading to Eq. (5.24) yielding optimality of ϕ'. This finishes the proof.

5.3.6 Tauberian Theorem

This section develops a Tauberian theorem that is used to provide the necessary ingredients for proving Theorem 5.7. This theorem is the continuous time counterpart of Theorem A.4.2 in Sennott [44]. A related assertion can be found in [27, Proposition A.5], however, in a weaker variant (without the Karamata implication, see Theorem 5.8, implication $(i) \implies (iii)$). The continuous time version seems deducible from Chap. 5 of the standard work on this topic [55]. We give a direct proof here.

Let $s : [0,\infty) \to \Re$ be a function that is bounded below by $-L$ say and $(\mathcal{B}([0,\infty)),\mathcal{B})$-measurable, where $\mathcal{B}$ denotes the Borel-σ-algebra on $\Re$, and $\mathcal{B}([0,\infty))$ the Borel-σ-algebra on $[0,\infty)$. Assume for any $\alpha > 0$ that

$$S(\alpha) = \int_{t=0}^{\infty} s(t)e^{-\alpha t}dt < \infty.$$

Furthermore, assume for any $T > 0$ that

$$S_T = \int_{t=0}^{T} s(t)dt < \infty.$$

Lemma 5.8. *Suppose that $L = 0$, i.e. s is a nonnegative function. Then for all $\alpha > 0$ it holds that*

$$\frac{1}{\alpha}S(\alpha) = \int_{T=0}^{\infty} e^{-\alpha T}S_T dT. \tag{5.28}$$

Furthermore, for all $\alpha > 0$, and $U \geq 0$ the following inequalities hold true:

$$\alpha S(\alpha) \geq \inf_{T\geq U}\left\{\frac{S_T}{T}\right\}\left(1-\alpha^2\int_{T=0}^{U} Te^{-\alpha T}dT\right), \tag{5.29}$$

and

$$\alpha S(\alpha) \leq \alpha^2\int_{T=0}^{U} e^{-\alpha T}S_T dT + \sup_{T\geq U}\left\{\frac{S_T}{T}\right\}. \tag{5.30}$$

Proof. We first prove Eq. (5.28). To this end, let $\alpha > 0$. Then,

$$\begin{aligned}
\frac{1}{\alpha}S(\alpha) &= \int_{u=0}^{\infty} e^{-\alpha u}du\int_{t=0}^{\infty} s(t)e^{-\alpha t}dt \\
&= \int_{t=0}^{\infty}\int_{u=0}^{\infty} s(t)e^{-\alpha(u+t)}dudt \\
&= \int_{T=0}^{\infty}\int_{t=0}^{T} s(t)e^{-\alpha T}dt\,dT \\
&= \int_{T=0}^{\infty} e^{-\alpha T}\int_{t=0}^{T} s(t)dt\,dT \\
&= \int_{T=0}^{\infty} e^{-\alpha T}S_T dT.
\end{aligned}$$

Interchange of integrals, change of variables are allowed, since the integrands are non-negative and the integrals are finite.

Next, we prove Eq. (5.29). To this end, we use Eq. (5.28). Then, for all $\alpha > 0$, $U \geq 0$

$$\begin{aligned}
\alpha S(\alpha) &= \alpha^2 \int_{T=0}^{\infty} S_T e^{-\alpha T} dT \\
&\geq \alpha^2 \int_{T=U}^{\infty} \frac{S_T}{T} T e^{-\alpha T} dT \\
&\geq \alpha^2 \inf_{t \geq U} \left\{ \frac{S_T}{T} \right\} \int_{T=U}^{\infty} T e^{-\alpha T} dT \\
&= \alpha^2 \inf_{t \geq U} \left\{ \frac{S_T}{T} \right\} \left(\int_{T=0}^{\infty} T e^{-\alpha T} dT - \int_{T=0}^{U} T e^{-\alpha T} dT \right) \\
&= \inf_{T \geq U} \left\{ \frac{S_T}{T} \right\} \left(1 - \alpha^2 \int_{T=0}^{U} T e^{-\alpha T} dT \right).
\end{aligned}$$

The first inequality uses explicitly that the integrand is non-negative.

Similarly, we expand from Eq. (5.28) to get Inequality (5.30) as follows. Let $\alpha > 0, U \geq 0$. Then,

$$\begin{aligned}
\alpha S(\alpha) &= \alpha^2 \int_{T=0}^{\infty} e^{-\alpha T} S_T dT \\
&= \alpha^2 \int_{T=0}^{U} e^{-\alpha T} S_T dT + \alpha^2 \int_{T=U}^{\infty} e^{-\alpha T} T \frac{S_T}{T} dT \\
&\leq \alpha^2 \int_{T=0}^{U} e^{-\alpha T} S_T dT + \sup_{T \geq U} \left\{ \frac{S_T}{T} \right\} \alpha^2 \int_{T=U}^{\infty} T e^{-\alpha T} dT \\
&\leq \alpha^2 \int_{T=0}^{U} e^{-\alpha T} S_T dT + \sup_{T \geq U} \left\{ \frac{S_T}{T} \right\} \alpha^2 \int_{T=0}^{\infty} T e^{-\alpha T} dT \\
&= \alpha^2 \int_{T=0}^{U} e^{-\alpha T} S_T dT + \sup_{T \geq U} \left\{ \frac{S_T}{T} \right\}.
\end{aligned}$$

Let $f : [0,1] \to \Re$ be an integrable function, and define

$$S_f(\alpha) = \int_{t=0}^{\infty} e^{-\alpha t} f(e^{-\alpha t}) s(t) dt.$$

Lemma 5.9. *Assume that* $L = 0$ *and*

$$W := \liminf_{\alpha \downarrow 0} \alpha S(\alpha) = \limsup_{\alpha \downarrow 0} \alpha S(\alpha) < \infty.$$

Let $r : [0,1] \to \Re$ *be given by*

$$r(x) = \begin{cases} 0 & x < 1/e \\ 1/x & x \geq 1/e. \end{cases}$$

Then

$$\lim_{\alpha\downarrow 0}\alpha S_r(\alpha)=\left(\int_{x=0}^{1}r(x)dx\right)\lim_{\alpha\downarrow 0}\alpha S(\alpha). \tag{5.31}$$

Proof. We first prove Eq. (5.31) for polynomial functions, then for continuous functions and finally for r. To show that Eq. (5.31) holds for polynomials, it is sufficient to prove it for $p(x)=x^k$. Thus,

$$\begin{aligned}\alpha S_p(\alpha)&=\alpha\int_{t=0}^{\infty}e^{-\alpha t}(e^{-\alpha t})^k s(t)dt\\&=\frac{1}{k+1}\left[\alpha(k+1)\int_{t=0}^{\infty}e^{-\alpha(k+1)t}s(t)dt\right]\\&=\int_{x=0}^{1}p(x)dx\left[\alpha(k+1)S(\alpha(k+1))\right].\end{aligned}$$

Taking the limit of $\alpha\downarrow 0$ proves Eq. (5.31) for polynomials. This is allowed because W is finite. Next we show Eq. (5.31) for continuous functions. The Weierstrass approximation theorem (see [3, 49, Sect. 13.33]) yields that a continuous function q on a closed interval can be arbitrary closely approximated by polynomials. Let p such that $p(x)-\varepsilon\le q(x)\le p(x)+\varepsilon$ for $0\le x\le 1$. Then, writing $p-\varepsilon$ for the polynomial $x\mapsto p(x)-\varepsilon$,

$$\int_{x=0}^{1}p(x)dx-\varepsilon\le\int_{x=0}^{1}q(x)dx\le\int_{x=0}^{1}p(x)dx+\varepsilon.$$

$$\begin{aligned}S_{p-\varepsilon}(\alpha)&=\int_{t=0}^{\infty}e^{-\alpha t}(p(e^{-\alpha t})-\varepsilon)s(t)dt\\&=\int_{t=0}^{\infty}e^{-\alpha t}p(e^{-\alpha t})s(t)dt-\varepsilon\int_{t=0}^{\infty}e^{-\alpha t}s(t)dt\\&=S_p(\alpha)-\varepsilon S(\alpha).\end{aligned}$$

This implies

$$0\le S_{p+\varepsilon}(\alpha)-S_{p-\varepsilon}(\alpha)\le 2\varepsilon S(\alpha),$$

As ε approaches 0, finiteness of W yields the result for continuous functions. In a similar manner r can be approximated by continuous functions q,q' with $q'\le r\le q$ as follows

$$q(x)=\begin{cases}0 & x<\frac{1}{e}-\delta\\ \frac{e}{\delta}x+e-\frac{1}{\delta}\ \frac{1}{e}-\delta\le x<\frac{1}{e}\\ \frac{1}{x} & x\ge\frac{1}{e},\end{cases}$$

$$q'(x)=\begin{cases}0 & x<\frac{1}{e}\\ \frac{e}{\gamma+\gamma^2 e}x-\frac{1}{\gamma+\gamma^2 e}\ \frac{1}{e}\le x<\frac{1}{e}+\gamma\\ 1/x & x\le\frac{1}{e}+\gamma.\end{cases}$$

This proves Eq. (5.31).

Theorem 5.8. *The following assertions hold.*

1. $\liminf_{T\to\infty} \frac{S_T}{T} \overset{(a)}{\leq} \liminf_{\alpha\downarrow 0} \alpha S(\alpha) \overset{(b)}{\leq} \limsup_{\alpha\downarrow 0} \alpha S(\alpha) \overset{(c)}{\leq} \limsup_{T\to\infty} \frac{S_T}{T}$;
2. *the following are equivalent*

$$\begin{aligned} &i)\ \liminf_{\alpha\downarrow 0} \alpha S(\alpha) = \limsup_{\alpha\downarrow 0} \alpha S(\alpha) < \infty; \\ &ii)\quad \liminf_{T\to\infty} \frac{S_T}{T} = \limsup_{T\to\infty} \frac{S_T}{T} < \infty; \\ &iii)\ \lim_{\alpha\downarrow 0} \alpha S(\alpha) = \lim_{T\to\infty} \frac{S_T}{T} < \infty. \end{aligned}$$

Proof. This proof is based on Sennott [44]. Clearly inequality (b) holds. So this leaves to prove inequalities (a) and (c).

Proof of inequality (a). First notice, that if we take $s \equiv M$ a constant function, then

$$\liminf_{T\to\infty} \frac{S_T}{T} = \liminf_{\alpha\downarrow 0} \alpha S(\alpha) = \limsup_{\alpha\downarrow 0} \alpha S(\alpha) = \limsup_{T\to\infty} \frac{S_T}{T}.$$

Therefore adding a constant M to the function s does not influence the result. Hence, it is sufficient to prove the theorem for nonnegative functions s. This means that the assumptions of Lemma 5.8 hold and we may use Inequality (5.29). Thus,

$$\inf_{T\geq U} \left\{ \frac{S_T}{T} \right\} \left(1 - \alpha^2 \int_{T=0}^{U} T e^{-\alpha T} dT \right) \leq \alpha S(\alpha).$$

Notice that $\lim_{\alpha\downarrow 0} \alpha^2 \int_{T=0}^{U} T e^{-\alpha T} dT = 0$, hence taking the liminf as $\alpha \downarrow 0$ gives

$$\inf_{T\geq U} \frac{S_T}{T} \leq \liminf_{\alpha\downarrow 0} \alpha S(\alpha).$$

Now taking the limit $U \to \infty$ on both sides gives

$$\liminf_{T\to\infty} \frac{S_T}{T} \leq \liminf_{\alpha\downarrow 0} \alpha S(\alpha),$$

which yields the result. Using Inequality (5.30) of Lemma 5.8 and applying the same reasoning proves inequality (c). Next we prove part 2. Part 1 implies that $(iii) \Longleftrightarrow (ii) \Longrightarrow (iii)$. So it is sufficient to prove that $(i) \Longrightarrow (iii)$ (Fig. 5.3).

Assume that (i) holds, then we may invoke Lemma 5.9. First notice that $\int_{x=0}^{1} r(x)dx = 1$, hence Eq. (5.31) reduces to

$$\lim_{\alpha \downarrow 0} \alpha S_r(\alpha) = \lim_{\alpha \downarrow 0} \alpha S(\alpha).$$

Moreover,

$$\alpha S_r(\alpha) = \alpha \int_{t=0}^{\infty} e^{-\alpha t} s(t) e^{\alpha t} \mathbb{1}_{\{e^{-\alpha t} \geq e^{-1}\}} dt = \alpha \int_{t=0}^{1/\alpha} s(t)dt = \alpha S_{1/\alpha}$$

To complete the proof, we have

$$\lim_{\alpha \downarrow 0} \alpha S(\alpha) = \lim_{\alpha \downarrow 0} \alpha S_r(\alpha) = \lim_{\alpha \downarrow 0} \alpha S_{1/\alpha} = \lim_{T \to \infty} \frac{S_T}{T}.$$

Acknowledgements We would like to thank Sandjai Bhulai for introducing us to the illustrative tandem queue model in Sect. 5.2.2. Moreover, he provided us with numerical results for Figs. 5.2 and 5.3.

Appendix: Notation

S & i, j, w, z	State space and states
Φ	Parameter space modelling decision rules and perturbations
ϕ	Elements of parameter space
δ	Decision rule and stationary, deterministic policy
$\delta(i)$	Decision rule in state i
$\mathcal{A}(i)$	Action set in state i
$\mathcal{A} = \prod_{i \in S} \mathcal{A}(i)$	Action space, equivalently the collection of all stationary, deterministic policies; equal to Φ (in this work) if no perturbations/truncations are modelled
α	Discount factor in continuous and discrete time
P^{ϕ}	One step transition matrix under parameter ϕ
$p^{\phi}(j \mid i)$	Transition probability to state j, when in state i and parameter ϕ chosen
Q^{ϕ}	Generator of Markov process operating under parameter ϕ, assumed conservative and stable
$q^{\phi}(j \mid i)$	Transition rate from state i to state j under parameter ϕ
$q^{\phi}(i)$	The total transition rate parameter of the exponentially distributed sojourn time in state i under parameter ϕ
P_t^{ϕ}	Transition matrix at time t under parameter ϕ
$p_t^{\phi}(j \mid i)$	Transition probability to state j at time t, when in state i and parameter ϕ is chosen
c^{ϕ}	Expected one step cost under parameter ϕ in discrete time, expected cost rate under parameter ϕ in continuous time

V_α^ϕ	α-Discount value function under parameter ϕ
V_α^*	Optimal α-discount value function
$V_{\alpha,n}$	Optimal expected α-discounted total cost up to (and including) time n
$V_\alpha^{*,(N)}$	Optimal α-discount value function for the N-perturbed MDP
$V_{\alpha,n}^{(N)}$	Optimal expected α-discounted total cost up to (and including) time n for the N-perturbed MDP
g^ϕ	Average expected cost under parameter ϕ
g^*	Optimal average expected cost
V_n	Optimal expected total cost up to (and including) time n
H^ϕ	Bias under parameter ϕ
H^*	Bias under an optimal policy
E_ϕ	Expectation operator under parameter ϕ
P_ϕ	Probability operator under parameter ϕ
$\mathbf{e}$	Vector of appropriate size consisting of ones
$\mathbf{I}$	Identity matrix of appropriate dimension
τ_z	Entrance time of state z, in discrete and continuous time
m_z^ϕ	Expected entrance time of state z, under parameter ϕ, in discrete and continuous time
$C_z^\phi(i)$	Expected total cost before reaching state z, under parameter ϕ, in discrete and continuous time

References

1. I.J.B.F. Adan, V.G. Kulkarni, A.C.C. van Wijk, Optimal control of a server farm. INFOR **51**(4), 241–252 (2013)
2. E. Altman, A. Hordijk, F.M. Spieksma, Contraction conditions for average and α-discount optimality in countable Markov games with unbounded rewards. Math. Oper. Res. **22**, 588–619 (1997)
3. T.M. Apostol, *Mathematical Analysis* (Addison Wesley Publishing Company, 1974)
4. Y. Aviv, A. Federgruen, The value iteration method for countable state Markov decision processes. Oper. Res. Lett. **24**, 223–234 (1999)
5. R. Bellman, A Markovian decision process. Technical report, DTIC Document (1957)
6. S. Bhulai, G.M. Koole, On the structure of value functions for threshold policies in queueing models. J. Appl. Prob. **40**(3), 613–622 (2003)
7. S. Bhulai, F.M. Spieksma, On the uniqueness of solutions to the Poisson equations for average cost Markov chains with unbounded cost functions. Math. Meth. Oper. Res. **58**(2), 221–236 (2003)

8. S. Bhulai, A.C. Brooms, F.M. Spieksma, On structural properties of the value function for an unbounded jump Markov process with an application to a processor sharing retrial queue. Queueing Syst. **76**(4), 425–446 (2014)
9. S. Bhulai, H. Blok, F.M. Spieksma, Competing queues with customer abandonment: optimality of a generalised cμ-rule by the smoothed rate truncation method. Technical report, Mathematisch Instituut Leiden (2016)
10. P. Billingsley, *Convergence of Probability Measures*. Wiley Series in Probability and Statistics, 2nd edn. (Wiley, New York, 1999)
11. H. Blok, Markov decision processes with unbounded transition rates: structural properties of the relative value function. Master's thesis, Utrecht University, 2011
12. H. Blok, Unbounded-rate Markov Decision Processes and optimal control. Structural properties via a parametrisation approach. Universiteit Leiden (2016). http://pub.math.leidenuniv.nl/~spieksmafm/theses/blok.pdf
13. H. Blok, F.M. Spieksma, Countable state Markov decision processes with unbounded jump rates and discounted cost: optimality and approximations. Adv. Appl. Probab. **47**, 1088–1107 (2015)
14. H. Blok, F.M. Spieksma, Structural properties of the server farm model. Technical report, Mathematisch Instituut Leiden (2016, in preparation)
15. V.S. Borkar, *Topics in Controlled Markov Chains* (Longman Scientific & Technical, Harlow, 1991)
16. R. Dekker, Denumerable Markov decision chains, optimal policies for small interest rates. PhD thesis, Universiteit Leiden, 1985
17. R. Dekker, A. Hordijk, Average, sensitive and Blackwell optimal policies in denumerable Markov decision chains with unbounded rewards. Math. Oper. Res. **13**, 395–421 (1988)
18. R. Dekker, A. Hordijk, Recurrence conditions for average and Blackwell optimality in denumerable Markov decision chains. Math. Oper. Res. **17**, 271–289 (1992)
19. R. Dekker, A. Hordijk, F.M. Spieksma, On the relation between recurrence and ergodicity properties in denumerable Markov decision chains. Math. Oper. Res. **19**, 539–559 (1994)
20. C. Derman, *Finite State Markovian Decision Processes* (Academic, New York, 1970)
21. S. Doroudi, B. Fralix, M. Harchol-Balter, Clearing analysis on phases: exact limiting probabilities for skip-free, unidirectional, Quasi-Birth-Death processes (2016, submitted for publication)
22. D.G. Down, G. Koole, M.E. Lewis, Dynamic control of a single-server system with abandonments. Queueing Syst. **67**(1), 63–90 (2011)
23. G. Fayolle, V.A. Malyshev, M.V. Menshikov, *Constructive Theory of Countable Markov Chains* (Cambridge University Press, Cambridge, 1995)
24. E.A. Feinberg, Total reward criteria, in *Handbook of Markov Decision Processes*, ed. by E.A. Feinberg, A. Shwartz. International Series in Operations Research and Management Science, vol. 40, chap. 5 (Kluwer Academic Publishers, Amsterdam, 2002), pp. 155–189

25. L. Fisher, S.M. Ross, An example in denumerable decision processes. Ann. Math. Stat. **39**(2), 674–675 (1968)
26. F.G. Foster, On the stochastic matrices associated with certain queuing processes. Ann. Math. Stat. **24**(3), 355–360 (1953)
27. X.P. Guo, O. Hernández-Lerma, *Continuous-Time Markov Decision Processes*. Stochastic Modelling and Applied Probability, vol. 62 (Springer, Berlin, 2009)
28. X.P. Guo, O. Hernández-Lerma, T. Prieto-Rumeau, A survey of recent results on continuous-time Markov decision processes. TOP **14**, 177–261 (2006)
29. A. Hordijk, *Dynamic Programming and Markov Potential Theory*. Mathematical Centre Tracts, vol. 51 (C.W.I., Amsterdam, 1974)
30. A. Hordijk, Regenerative Markov decision models. Math. Program. Study **6**, 49–72 (1976)
31. A. Hordijk, P.J. Schweitzer, H.C. Tijms, The asymptotic behaviour of the minimal total expected cost for the denumerable state Markov decision model. J. Appl. Prob. **12**, 298–305 (1975)
32. M. Kitaev, Semi-Markov and jump Markov controlled models: average cost criterion. Theory Prob. Appl. **30**, 272–288 (1986)
33. G.M. Koole, *Monotonicity in Markov Reward and Decision Chains: Theory and Applications*, vol. 1 (Now Publishers Inc., Hanover, 2007)
34. S.A. Lippman, On dynamic programming with unbounded rewards. Manage. Sci. **21**, 1225–1233 (1975)
35. S.P. Meyn, R.L. Tweedie, *Markov Chains and Stochastic Stability* (Springer, Berlin, 1993)
36. A. Piunovskiy, Y. Zhang, Discounted continuous-time Markov decision processes with unbounded rates and history dependent policies: the dynamic programming approach. 4OR-Q J. Oper. Res. **12**, 49–75 (2014)
37. T. Prieto-Rumeau, O. Hernández-Lerma, Discounted continuous-time controlled Markov chains: convergence of control models. J. Appl. Probab. **49**(4), 1072–1090 (2012)
38. T. Prieto-Rumeau, O. Hernández-Lerma, *Selected Topics on Continuous-Time Controlled Markov Chains and Markov Games*. ICP Advanced Texts in Mathematics, vol. 5 (Imperial College Press, London, 2012)
39. M.L. Puterman, *Markov Decision Processes: Discrete Stochastic Programming*, 2nd edn. (Wiley, Hoboken, NJ, 2005)
40. S.M. Ross, Non-discounted denumerable Markovian decision models. Ann. Math. Stat. **39**(2), 412–423 (1968)
41. H.L. Royden, *Real Analysis*, 2nd edn. (Macmillan Publishing Company, New York, 1988)
42. L.I. Sennott, Average cost optimal stationary policies in infinite state Markov decision processes with unbounded costs. Oper. Res. **37**(4), 626–633 (1989)
43. L.I. Sennott, Valute iteration in countable state average cost Markov decision processes with unbounded costs. Ann. Oper. Res. **28**, 261–272 (1991)
44. L.I. Sennott, *Stochastic Dynamic Programming and the Control of Queueing Systems*. Wiley Series in Probability and Statistics (Wiley, New York, 1999)

45. R.F. Serfozo, An equivalence between continuous and discrete time Markov decision processes. Oper. Res. **27**(3), 616–620 (1979)
46. F.M. Spieksma, Geometrically ergodic Markov Chains and the optimal Control of Queues. PhD thesis, Leiden University, 1990. Available on request from the author
47. F.M. Spieksma, Countable state Markov processes: non-explosiveness and moment function. Prob. Eng. Inf. Sci. **29**(4), 623–637 (2015)
48. R.E. Strauch, Negative dynamic programming. Ann. Math. Stat. **37**(4), 871–890 (1966)
49. E.C. Titchmarsh. *The Theory of Functions*, 2nd edn. (Oxford University Press, Oxford, 1986)
50. R.L. Tweedie, Criteria for ergodicity, exponential ergodicity and strong ergodicity of Markov processes. J. Appl. Prob. **18**, 122–130 (1981)
51. N.M. van Dijk, On uniformization of time-inhomogeneous continuous-time Markov chains. Oper. Res. Lett. **12**, 283–291 (1992)
52. N.M. van Dijk, Error bounds and comparison results: the Markov reward approach, in *Queueing Networks. A Fundamental Approach*, ed. by R.J. Boucherie, N.M. van Dijk. International Series in Operations Research and Management Science, chap. 9 (Springer US, 2011), pp. 379–461
53. R.R. Weber, S. Stidham Jr., Optimal control of service rates in networks of queues. Adv. Appl. Prob. **19**(1), 202–218 (1987)
54. J. Wessels, Markov programming by successive approximations with respect to weighted supremum norms. J. Math. Anal. Appl. **58**(2), 326–335 (1977)
55. D.V. Widder, *The Laplace Transform* (Princeton University Press, Princeton, 1946)

Part II
Healthcare

Chapter 6
Markov Decision Processes for Screening and Treatment of Chronic Diseases

Lauren N. Steimle and Brian T. Denton

Abstract In recent years, Markov decision processes (MDPs) and partially observable Markov decision processes (POMDPs) have found important applications to medical decision making in the context of prevention, screening, and treatment of diseases. In this chapter, we provide a review of state-of-the-art models and methods that have been applied to chronic diseases. We provide a tutorial about how to formulate and solve these important problems emphasizing some of the challenges specific to chronic diseases such as diabetes, heart disease, and cancer. Then, we illustrate important considerations for model formulation and solution methods through two examples. The first example is an MDP model for optimal control of drug treatment decisions for managing the risk of heart disease and stroke in patients with type 2 diabetes. The second example is a POMDP model for optimal design of biomarker-based screening policies in the context of prostate cancer. We end the chapter with a discussion of the challenges of using MDPs and POMDPs for medical contexts and describe some important future directions for research.

6.1 Introduction

Chronic diseases are the leading cause of death and disablement in many countries [1]. Although these diseases cannot be cured, they can be controlled by screening and treatment. Clinicians are tasked with deciding which screening and treatment options are most beneficial for a patient. These decisions are made sequentially

L.N. Steimle (✉) • B.T. Denton
Department of Industrial and Operations Engineering, University of Michigan, 1205 Beal Ave, Ann Arbor, MI 48109, USA
e-mail: steimle@umich.edu; btdenton@umich.edu

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*, International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_6

over long periods of a patient's life and are made in an uncertain environment. Although clinicians can observe a patient's current test results, there is uncertainty in the future progression of the disease, the effect of treatment on the patient, and even the correctness of test results. Medical decisions have grown even more complicated due to the aging patient population. Older patients often have multiple chronic conditions, and treatment for one condition may worsen another. Health care providers have recognized these growing problems and have responded with increased expenditures on data collection and tracking systems. With the growth in medical data comes the need for analytical methodology to convert these data into information. Recently, operations research methods have proven to be powerful tools to analyze these data to guide screening and treatment decisions.

Markov decisions processes (MDPs) are increasingly being used in the analysis of medical decisions, especially chronic disease screening and treatment decisions. Both screening and treatment decisions are characterized by large state spaces that define the severity of the disease, patient-specific clinical risk factors, and medication histories, and these decisions have uncertain outcomes due to differences among patients such as genetic, environmental, and dietary factors. The framework of MDPs lends itself well to these decisions since they are made sequentially over time in a fundamentally stochastic environment. Further, partially observable MDPs (POMDPs) are useful for studying systems in which the true state of the system is not known exactly, which is usually the case when screening for a chronic disease.

Modeling screening and treatment decisions using MDPs is not without its challenges. These clinical decisions take place over long time horizons (sometimes decades) under constraints due to medication conflicts, clinical practice requirements, or budget constraints. Furthermore, the influence of a patient's treatment and screening history on future decisions leaves these models subject to the *curse of dimensionality* due to the dramatic increase in the size of the state space that can be caused by this history dependence. As a result, optimization of the stochastic and sequential decision making process gives rise to computationally-intensive problems that are difficult to solve, even with state-of-the-art algorithms and computing resources. Fortunately, many of these problems have promising structural properties that can be exploited to achieve meaningful theoretical insights and lead to efficient exact and/or approximation methods.

The remainder of this chapter is organized as follows: in Sect. 6.2, we discuss some applications of MDPs to chronic diseases. In Sect. 6.3, we discuss how to formulate an MDP/POMDP model in the context of chronic disease and solution methods that can be used to determine optimal policies for these models. In Sects. 6.4 and 6.5, we give in-depth descriptions of an MDP used for the treatment of type 2 diabetes and a POMDP model used for screening of prostate cancer, respectively. We end the chapter with discussion of the open challenges that need to be addressed when using MDP/POMDP models for chronic diseases and some concluding remarks.

6.2 Background on Chronic Disease Modeling

Surveys of operations research applications in healthcare can be found in [2–4]. Many of the examples are in the context of healthcare operations management, which has been an important application area for decades. In contrast to operations management, the study of disease screening and treatment policies has a shorter history and is confined to a relatively small, but fast growing, number of topic areas including liver and kidney transplant decisions [5–11], breast cancer screening [12, 13], intensity modulated radiation therapy [14–16] and brachy-therapy [17] for cancer treatment, the treatment of HIV [18], and public policy decisions related to the transmission of communicable diseases [19, 20].

MDPs can be used to study sequential decisions made in uncertain environments, which is why they are so powerful for the analysis of chronic disease screening and treatment. Before describing how these models are formulated, we provide some motivation for the study of MDPs in the context of chronic diseases by giving the following examples of clinically-relevant questions that have been answered:

- *At what point should a patient with HIV initiate highly active antiretroviral therapy (HAART)?*
  Human Immunodeficiency Virus (HIV) is a virus that attacks the CD4 white blood cells to the point the body can no longer protect itself against infections and disease. Acquired Immune Deficiency Syndrome (AIDS) is caused by HIV and eventually leads to death. Once someone acquires HIV, the virus will remain in the body for the remainder of that person's life. Highly active antiretroviral therapy (HAART) prevents the virus from multiplying and is the standard treatment for HIV patients. However, it was debated whether to "hit early, hit hard" with HAART, as was the treatment model in the late 1990s, or to wait until the CD4 count falls between 200 and 350 as suggested by more recent guidelines. The authors of [18] used an infinite-horizon MDP with the objective of maximizing a patient's total expected lifetime or quality-adjusted lifetime to answer this open question. The states of the MDP were defined by a patient's CD4 count, and at each monthly decision epoch, the decision was to *"initiate HAART"* or *"wait to initiate HAART"*. The authors proved that the optimal policy prescribes initiating therapy if and only if the CD4 count falls below a certain threshold. The optimal policy suggested that HAART should be initiated earlier supporting the "hit early, hit hard" approach to HIV treatment.

- *When should women receive mammograms to screen for breast cancer?*
  Breast cancer is the second leading cause of cancer death for women in the United States [21]. Detecting breast cancer in its early stages allows for treatment and decreases the risk of a breast cancer mortality. A mammogram is an X-ray image of the breast that can be used to detect breast cancer before a woman develops symptoms. If a mammogram shows a suspicious area, a biopsy can be performed to determine if the abnormality is cancer. While these tests are useful in determining if a patient has cancer, they are not perfect.

Mammograms can lead to radiation exposure and pain, and biopsies are an invasive procedure associated with pain and anxiety. Further, mammograms can give false negative and false positive results. The authors of [22] created a finite-horizon POMDP model to determine personalized mammography screening policies that depend on a woman's personal risk factors and past screening results. The unobservable states represent which stage of cancer the patient has such as no cancer, noninvasive cancer, invasive cancer, invasive cancer under treatment, or death. The actions of this POMDP are *"wait"* and *"mammography"*. If the action chosen is mammography, the decision maker can observe a positive or negative mammogram result. If the action is to wait, the patient can give a self-detection result that is either positive or negative. If a mammogram is positive, the patient will get a biopsy, and if a self-detection is positive, the patient will get a mammogram. With these observations in mind, the decision maker can update her *belief state* which describes the probability that a patient is in any given state given the history of mammogram results. The authors find that a control-limit policy exists that depends on the risk of noninvasive and invasive cancers and that a patient's screening history may affect the decision of whether to perform a mammography or not.

- *When should a patient with end-stage liver disease accept a living-donor transplant?* For patients with end-stage liver diseases such as primary biliary cirrhosis, hepatitis C, and acute failure (fulminants) disease, organ transplantation is the only treatment option. Provided that a patient with end-stage liver disease has a willing living donor, it might seem the patient should receive a transplant as soon as possible. However, depending on the quality of the match with the donor and the current health of the patient, this decision might give a lower expected total lifetime for the patient compared with the decision to wait. To analyze this situation, the authors of [5] create an infinite-horizon MDP model in which the state space is represented by a patient's "Model For End-Stage Liver Disease"(MELD) score. The MELD score quantifies the severity of end-stage liver disease based on laboratory results and is used for the purpose of transplant decisions. Higher MELD scores are associated with more severe liver disease. At each daily decision epoch, the actions are *"transplant"* and *"wait"*. If the decision is to wait, the patient will receive a reward of one life day and then progress probabilistically among the health states or die. Once the decision to transplant is made, the patient transitions into an absorbing state and receives a reward corresponding to the expected life days associated with the health of the patient at the time of the transplantation and the quality of the match with the donor. The authors prove that the optimal policy has a control-limit structure in which the patient will only accept a liver of a given quality if her MELD score is worse than the control-limit. For example, a MELD score of 20 is the control-limit given that the quality of the match has a score of 4. Therefore, a patient with a MELD score of 25 should accept this liver to transplant while a patient with a MELD score of 15 should wait to transplant.

These examples illustrate some treatment and screening decisions that can be analyzed using MDPs. More examples of MDPs used in medicine can be found in the reviews by the authors of [23, 24]. This chapter differs from these previous reviews in that we provide an in-depth discussion of how to formulate MDP models for chronic disease screening and treatment problems. We also provide detailed examples that illustrate MDP model formulation, validation, solutions, and interpretation of results. Finally we compare and contrast perfectly observable and imperfectly observable contexts. With this motivation, we will proceed to more formally describe how MDPs can be formulated to generate insights for screening or treating a chronic disease.

6.3 Modeling Framework for Chronic Diseases

The remainder of this chapter will focus on the modeling framework for MDPs specifically in the context of screening and treatment applications. This section will provide a tutorial on how to formulate, solve, and validate these models. In the following sections, we will provide several examples to illustrate the development of the formulation and potential challenges faced by researchers.

6.3.1 MDP and POMDP Model Formulation

To build an MDP model of a chronic disease treatment process, one must define the *decision epochs*, *time horizon*, *state space*, *action space*, *transition probabilities*, and *rewards* as they relate to the specific disease and screening/treatment options being considered.

Decision Epochs: Treatment and screening decisions are made at each decision epoch. The length of time between decision epochs for a chronic disease model usually corresponds to the time between treatment and/or screening decisions made by the clinician. For instance, in the case of liver transplantation, decisions about whether to transplant or not could be made daily, while in the case of type 2 diabetes, decisions about which medications to initiate are more likely to be made less frequently (e.g. every 6 or 12 months based on clinical guidelines). Determining the ideal time interval requires some understanding of the disease context and clinical practice.

Time Horizon: Another modeling choice is whether to consider a finite-horizon formulation, in which there are a finite number of decision epochs, or an infinite-horizon formulation. While the patient will die in a finite amount of time, some researchers use an infinite-horizon approach for treatment decisions when the time between epochs is short relative to the length of the horizon over which decisions

are made. For example, in organ transplantation, if the decision epochs are daily, it may be a suitable approximation to use an infinite-horizon. Usually infinite-horizon problems are associated with an absorbing state that is reached with probability 1, such as a post-treatment absorbing state. Moreover, infinite-horizon models are often *stationary*, i.e., model parameters do not vary over time.

State Space: The state space of the model represents the information that would be useful to a clinician when making decisions regarding a patient. A state vector typically includes the patient's health status, demographic information, and relevant medical history.

A patient's health status is usually defined by a number of clinical risk factors or a risk score that can be used by clinicians to predict the severity of a disease or the likelihood of developing a disease. For example, when determining whether or not to transplant a liver, clinicians consider a patient's MELD score which depends on a number of laboratory values that are useful in determining the severity of liver disease. While MELD scores are integer-valued, other metabolic risk factors, such as body mass index (BMI), are continuous. Most MDP models used for medical decisions discretize the true continuous state space to reduce the computation needed to solve the model. A finer discretization may be more representative of the true continuous state space, but it also increases the size of the state space and therefore the computation required to solve the model. Further, a finer discretization will decrease the number of observed transitions for some state-action pairs introducing more sampling error into the estimates of the transition probabilities. Reference [25] provides a discussion of the trade-off between the model error introduced with a more coarse discretization and the sampling error that is associated with a finer discretization.

A patient's demographic information can be important for defining the state space of a model. The dynamics of some diseases vary depending on the demographics of the patient such as age and race. For example, [12] considers age because older women are at higher risk for developing breast cancer, but breast cancer is less aggressive in these women. These dynamics may be important in determining the optimal treatment or screening policies, but incorporating this information might require formulation and validation of unique models for these different populations.

Information about a patient's medical history, such as medication history or history of adverse events, may affect treatment decisions. For example, once a patient has had one heart attack, she is at increased risk to have a second heart attack. Although this history is important, MDP models require that the transitions among states must maintain the Markov property, i.e, the next state may only depend on the current state and the action taken. To maintain this property, it is necessary to incorporate any necessary history of the patient into the state definition. For example, the state definition may include which medications a patient has already initiated or how many adverse events the patient has already had.

In most MDP models of chronic disease, there is an absorbing state representing major complication and/or death. In some models, there are separate death states depending on the cause of death (e.g. death from a heart attack, death from other

causes). It may be necessary to use more than one absorbing state when absorbing states that are reachable from a given health state vary or when rewards vary depending on the absorbing state that is reached. Defining the state space is closely tied to what sources exist to estimate transition probabilities, such as statistical survival models or patient data.

POMDPs are a generalization of MDPs in which the decision maker does not know the state of the system with certainty. This generalization is particularly useful within the context of chronic disease, because often clinicians cannot be 100% sure of the health state of their patients. While screening and diagnostic tests provide valuable information, these tests sometimes give false positive and false negative test results which leaves the true health state of the patient uncertain. In a POMDP, the state space is defined by a *core process* and an *observation process* (also referred to as a *message process*). With respect to chronic diseases, the core process corresponds to the true health of a patient, such as cancer-free, has non-invasive cancer, has invasive cancer, in treatment, or dead. To a clinician, the first three states are unobservable, meaning that the clinician cannot know with certainty the true state of the patient. The observation process corresponds to observable test results, such as a mammogram. The core process and the observation process are tied together probabilistically through an *information matrix* with elements that define probabilities of a particular observation given a particular core state. For example, the decision maker may know the true and false positive and negative rates of a biopsy based on clinical studies. Using Bayesian updating, the relationship between the core and observation processes and the observed test result can be used to continually update a *belief state* over the time horizon of the POMDP. The belief state is a probability distribution describing the believed true state of the system based on the decision maker's past observations. For additional details specific to POMDPs, the reader is referred to [26, 27].

Action Space: To identify the action space of the MDP, one must identify which screening or treatment options to consider. In the case where there is a clear "best" treatment option, the action space might be only two actions: treat the patient with the best therapy or wait. These are typically referred to as *optimal stopping-time problems* in the literature, because the decision maker aims to choose the optimal time to stop the process and enter the absorbing post-treatment state. For instance, deciding when to transplant an organ is usually a stopping-time problem with the action space being transplant or wait to transplant.

For some diseases, it is not clear which therapy is the best or how different therapies should be used together to treat the patient. In these cases, the action space can grow quite large because of the combinatorial nature of the actions. For example, if $M = \{m_1, m_2, \ldots, m_n\}$ is a set of different drugs that can be used in any combination to treat a patient, the size of the action space is $2^{|M|}$ and thus grows exponentially in the number of treatments considered.

In a POMDP model, the decision maker can take actions to gain information about the state of the system. For example, screening decisions can be modeled using POMDP models where the action space might represent the different types

of screening tests available. Performing a screening test may not change the natural progression of the disease, but it can provide the decision maker with valuable information about the true health state of the patient, which in turn may be used to decide whether to do more invasive testing such as biopsy or radiologic imaging.

Transition Probabilities: The transition probabilities in an MDP model of a chronic disease usually describes the progression of the disease with and without treatment, the probability of an adverse event, and the probability of death. To describe the progression of a disease, a key step is to build a *natural history model*. The natural history model describes how the disease progresses under no treatment. Creating this model can be challenging because medical records will only contain data about patients who have been diagnosed and treated for the disease. To build a natural history model, one can use longitudinal data to estimate the effects of treatment by observing measurements of risk factors before and after a patient starts the treatment. In this way, one could estimate how the disease would progress if no treatment was given to a patient. It is important to note that these measures can be affected by bias associated with patterns that influence which patients are referred for treatment. For example, patients who initiate blood pressure lowering medications would typically have higher than normal blood pressure and may exhibit greater relative reduction in blood pressure than the general population.

When there is a clear "best" therapy, as is the case in optimal stopping-time problems, the modeler is not concerned with the effect of treatment on the transition probabilities. Upon initiating treatment, the patient will transition to an absorbing state representing post-treatment with probability 1. In other cases, the modeler must consider how treatment affects the transition probabilities. Presumably, initiating treatment will lower the probability of having an adverse event or dying from the disease. A recent proliferation of statistical models for estimating the risk of chronic disease complications can provide these inputs for MDPs. For instance, statistical models for type 2 diabetes include: the *Framingham* model [28–30], the *UKPDS* model [31–34], and the *ACC/AHA* pooled risk calculator [35]. These models predict the probability of diabetes complications such as cardiovascular events (stroke and coronary heart disease), kidney failure, and blindness. Inputs include gender, race, family history, and metabolic factors like cholesterol, blood pressure, and blood glucose. Treatment can affect some of the inputs to these models and therefore can affect the transition probability to an adverse event state.

Another key input to an MDP model is the probability associated with transitioning to the death state. The probability of death caused by something other than the disease of interest is called *all other cause mortality*. All other cause mortality can have a large impact on treatment decisions. As all other cause mortality increases, treatment can become less beneficial since the probability of a complication or dying from the particular disease of focus for the MDP is not as likely. This is particularly important for chronic diseases that progress slowly. For example, the American Urology Association recommends not screening men for prostate cancer after age 75 because men who have not been diagnosed with prostate cancer by this age are not likely to die from this slowly progressing disease. Estimates for all other

cause mortality can typically be found using mortality tables from the Centers for Disease Control and Prevention (CDC) [36].

Rewards: The rewards and costs in a chronic disease MDP model may be associated with the economic and health implications associated with treatment and screening policies. To determine the relevant rewards and costs, one must identify the perspective of the decision maker: patient, third-party payer (e.g. Blue Cross Blue Shield, Medicare), or a societal perspective that combines these different perspectives. Treating or screening a patient for a chronic disease will offer some reward to the patient, such as a potentially longer life. However, these benefits come at some "cost" to the patient, whether it be a reduction in quality of life, such as side effects due to medication or discomfort due to a screening test, or a financial cost, such as medication or hospitalization expenses. Health services researchers typically use quality-adjusted life years (QALYs) to quantify the quality of a year of life with the discomfort due to medical interventions. A QALY of 1 represents a patient in perfect health with no disutility due to medical interventions and side effects of treatment. As the patient's quality of life decreases, whether from medication side effects or disablement from a disease, the patient's QALY value will tend towards zero. (The reader is referred to [37] for a review of QALYs and other quality of life measures.) Some MDP models are only concerned with maximizing a patient's QALYs. Other models take a societal perspective and attempt to balance the health benefits of treatment with the corresponding monetary costs of medical interventions. To balance these competing objectives, a common approach is to use a *willingness to pay* factor, which assigns a monetary value to a QALY. Values of $50,000 and $100,000 per QALY have commonly been used in the literature; however, the exact value to use is often debated [38].

MDPs are rather data-intensive due to the need for transition probabilities and rewards for each state-action pair. However, after gleaning these inputs from the literature or longitudinal patient data, solving these MDPs can generate meaningful insights into how and when to screen for and treat chronic diseases.

6.3.2 Solution Methods and Structural Properties

Various algorithms have been developed for solving MDPs. The appropriate method for solving an MDP depends on whether the MDP is formulated as an infinite-horizon or finite-horizon problem and the size of the state and action spaces. Methods such as *policy iteration*, *value iteration*, and *linear programming* have been used to solve infinite-horizon problems, while *backwards induction* is typically used to solve finite-horizon problems. One problem with MDP formulations is that they are subject to the curse of dimensionality. This is seen in MDPs for chronic disease where the size of the state space grows exponentially with the number of health risk factors defining the state. To circumvent this problem, approximation algorithms

can be used. There has been a great amount of research on approximate dynamic programming in general, but these approaches tend to be highly context dependent and very little work has been done in the context of chronic disease. References [39, 40] provide a thorough review of approximation methods of MDPs.

Many MDP models for chronic diseases have certain structural properties that can be exploited for computational gains. One such property is the *increasing failure rate* (IFR) property describing the transition probability matrices. In the context of chronic diseases, the IFR property means that the worse the health status of the patient is, the more likely that the health status will become even worse. Usually this ordering naturally follows the severity of the chronic disease, with the ordering of the states defined by a patient's health status. For certain optimal stopping-time problems, it has been shown that the IFR property together with some additional (and nonrestrictive) conditions guarantees an optimal threshold policy (see Chap. 4 of [41]). These conditions have be used in the context of HIV [18], liver disease [5], and type 2 diabetes [42] to prove the existence of an optimal *control-limit policy*. A control-limit policy is one in which one action is used for all states below a certain value (e.g. wait to transplant if the MELD score is below 25) and another action for all states above a certain value (e.g. transplant if the MELD score is at least 25). Proving the existence of a control-limit policy can decrease the computational effort required to solve the MDP model, since the value function does not need to be explicitly calculated for every state-action pair.

POMDPS are generally much more challenging to solve than MDPs. Early methodological studies focused on exact methods that exploit the fact that the optimal value function for a POMDP is convex, and in the finite-horizon case it is piecewise linear and expressible using a finite set of supporting hyperplanes. The first exact method was provided by Smallwood and Sondik [43]. The authors proposed an iterative approach to generate supporting hyperplanes at each decision epoch. Due to exponential growth in the number of hyperplanes with respect to the number of decision epochs and observations and the fact that many of the hyperplanes are dominated, the authors further proposed an approach to reduce the number of hyperplanes to a minimal set using a linear programming formulation to identify dominated hyperplanes. Many authors have built on this early approach by developing more efficient ways of pruning unnecessary hyperplanes, including *incremental pruning* [44] and the *witness method* [45]. Exact methods are generally limited to small POMDPs. A well-known approximation approach for moderate-sized POMDPs is based on discretizing the continuous belief state to obtain an approximate finite state MDP. One of the first proposed approaches was the *fixed-grid algorithm* proposed by Eckles [46]. Many enhancements, including variable grid based approaches have built on this early idea. The reader is referred to [47] for discussion of finite grid based approaches. Grid based methods are limited in their applicability to large-scale POMDPs. For this reason, it is often necessary to develop approximation methods tailored to particular applications.

6.3.3 Model Validation

Once an MDP has been solved, it is critical to determine whether the results of the model are valid. Below are some common ways to validate MDP models for chronic diseases.

Expert Opinion: After the MDP has been solved, one can seek the opinion of an expert in the field, such as a clinician or a health services researcher, to determine if the results of the model are realistic. This form of validation is not very strong since it is subjective. Some experts may have differing opinions of whether the model results are actually valid. However, this form of validation is probably the easiest to use and can be a first step in validating the model before turning to more objective procedures.

Independent Study: To validate an MDP, one could compare the results to a model developed independently. For instance, an alternative stochastic model could be compared to the MDP using a reference policy (e.g. an existing screening or treatment guideline.) This approach is particularly useful if there is an existing *gold standard* model to compare to.

Retrospective Validation: Retrospective validation compares the results of the MDP to past observed outcomes of an existing patient cohort. If this method of validation is used, one should use a different cohort for calibration of the model and for validation of the model. Using the same cohort to calibrate and validate the model could lead to *optimism bias*.

Prospective Validation: Prospective validation, the gold standard of validation, involves using the model to predict outcomes and comparing the predictions to the actual outcomes. This form of validation is considered very strong, because there is no contamination between data used to calibrate the model and the data used to validate it. However, the outcomes of interest in chronic disease modeling are long-term, which can lead to long periods of time between the obtainment of the results and the validation of the model. As a result, this form of validation is almost never done.

Validating the model is an important step to ensure that the results from the model are useful to clinicians. If the model cannot be validated, the modeler should carefully consider whether the assumptions of the model are justified, if the model parameters are accurate and generalizable to other patient populations, and if the model was implemented without errors. Sensitivity analysis often plays an important role in addressing concerns about inaccuracy of model parameters.

6.4 MDP Model for Cardiovascular Risk Control in Patients with Type 2 Diabetes

This section presents model formulation, solutions, and analysis of results for an MDP in the context of type 2 diabetes. Advances in medical treatments have extended the average lifespan of individuals and transformed many diseases from life-threatening in the near term to chronic conditions in need of long-term management. Diabetes is a good example. With 9.3% of the U.S. population estimated to have diabetes, it is recognized as a leading cause of mortality and morbidity [48]. The disease is associated with many serious complications such as coronary heart disease (CHD), stroke, blindness, kidney disease, limb amputation, and neurological disorders.

Patients with diabetes are at much higher risk of stroke and CHD events than those without diabetes. The risk of having one of these adverse events is affected by a number of risk factors including gender, race, height, weight, glucose, total cholesterol, high density lipids (HDL—often referred to as "good cholesterol"), and blood pressure (systolic and diastolic). Several medications now exist that can control cholesterol and blood pressure for patients with type 2 diabetes. However, there is considerable disagreement in the health care community about how best to use these medications [49–51]. Risk models exist to predict an individual patient's probability of complications related to diabetes [29–32]; but alone they cannot provide optimal treatment decisions. Further, these risk models often give conflicting estimates of patient's risk, which adds another challenge to the decision-making process.

Historically, guidelines for the treatment of cholesterol and blood pressure have been "one size fits all" guidelines that do not account for the different risk profiles of the heterogeneous population. The guidelines for cholesterol treatment and the guidelines for blood pressure treatment in the United States were created by two independent committees. This artificial separation of guidelines for treating risk factors that both influence the risk of CHD and stroke could potentially lead to over-treatment of patients and increases in medical costs. These issues provide great motivation for an MDP approach to treatment planning that combines decisions for cholesterol and blood pressure control.

Recently, MDPs have been used to study the optimal treatment of patients with type 2 diabetes. Kurt et al. [42] and Denton et al. [52] analyze the optimal time to initiate statins, the most common drug for managing cholesterol. Mason et al. [53] extends this work to study the effect of imperfect adherence on the optimal policy. Mason et al. [54] uses an MDP to determine the optimal simultaneous management of blood pressure and cholesterol. For the remainder of this section, we use the model in [54] as an example of model formulation, the effect of model parameters, and how the optimal compares to the guidelines. Additionally, we provide new results based on more recent data including a new risk model for estimating the probability of cardiovascular events [35].

6.4.1 MDP Model Formulation

In this MDP model, patients with type 2 diabetes progress between states defined by blood pressure and cholesterol levels. At every decision epoch, a clinician observes the patient's risk factors (i.e., cholesterol and blood pressure levels) and decides which medications (if any) to prescribe to the patient. This model takes a societal perspective and uses a bi-criteria objective, which balances the goal of having a low discounted medication cost with the goal of primary prevention (i.e. delaying the first occurrence of a CHD event or a stroke). Figure 6.1 gives a schematic representation of this decision process.

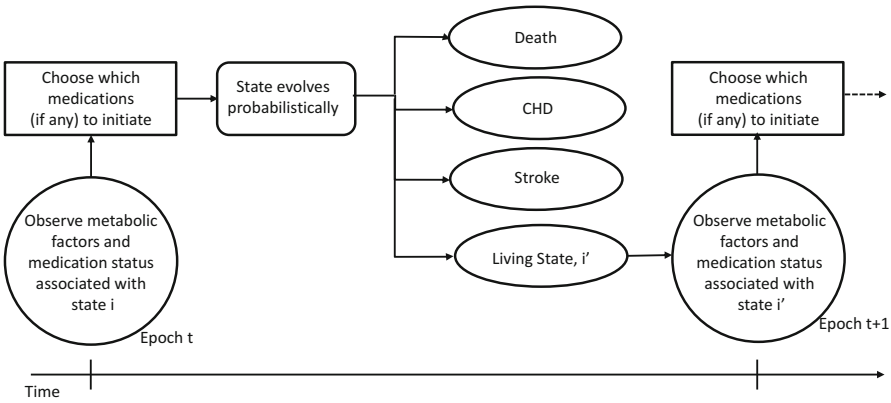


Fig. 6.1: The treatment decision process for managing cholesterol and blood pressure for patients with type 2 diabetes

Decision Epochs/Time Horizon: The decision of which medications to initiate is revisited periodically within a finite horizon with N (yearly) decision epochs, with non-stationary rewards and transition probabilities. The set of decision epochs is $\mathcal{T} = \{0, 1, 2, \ldots, N\}$. It is possible that some patients may live beyond decision epoch N and it is necessary to estimate the rewards that they will accrue in these additional years of life. To do this, an infinite-horizon approximation is used beyond epoch N in which treatment is held constant. This formulation is consistent with regular annual primary care visits for most adults. We assume $N = 100$ because physicians will not typically prescribe new medications to patients after they have reached a certain age.

State Space: The state space is made up of *living states* and *absorbing states*. The set of living states is denoted $S_{\mathcal{L}}$ and the states in this set are defined by a number of factors that characterize a patient's level of cardiovascular risk. Some of these

factors, such as metabolic levels and medication status, change over time. Because changes in these values affect the cardiovascular risk, it is important to incorporate these values into the state space. Other relevant information such as race and gender, is incorporated into the model through the transition probability and reward parameters.

When considering R metabolic factors and M medications, a living state is represented by a vector $\mathbf{s} = \{s_1, \ldots, s_R, s_{R+1}, \ldots, s_{R+M}\} \in S_{\mathcal{L}}$. In this model, the first R components of $\mathbf{s}$ correspond to measurements of a patient's total cholesterol, HDL, and systolic blood pressure, and each of the next M components are binary indicators specifying whether or not the patient is on the corresponding medication.

In practice, measurements of cholesterol and blood pressure are continuous. To create a discrete state space, these continuous values are discretized according to clinically-relevant thresholds and then labeled low (L), medium (M), high (H), and very high (V). For metabolic risk factor k, we have $s_k \in \{L, M, H, V\}$. The actual cutpoints for these states were based on a combination of clinical considerations and availability of data for estimating transition probabilities (please see [54] for a detailed discussion of this).

As stated in Sect. 6.3, MDPs must maintain the Markov property, and thus any necessary information from the past must be incorporated into the state space. In this model, it is necessary to know whether a patient is already on a medication or not, and therefore this information must be added to the state space. Consider the jth medication: if $s_{R+j} = 0$, the patient is not using medication j and if $s_{R+j} = 1$, the patient is using medication j. Notice that, in this model, the size of the living state space is $|S_{\mathcal{L}}| = 4^R \cdot 2^M$ and therefore the size of the living state space grows exponentially in R and M. Also, if a finer discretization of the metabolic risk factors was used, this growth would be even faster.

The model also has a set of absorbing states $S_{\mathcal{D}}$. These state vectors take on values that represent having a CHD event ($\mathbf{d}^C$), having a stroke ($\mathbf{d}^S$), or dying from a cause other than CHD or stroke ($\mathbf{d}^O$). The set of all absorbing states will be represented as $S_{\mathcal{D}} = \{\mathbf{d}^C, \mathbf{d}^S, \mathbf{d}^O\}$. Because primary prevention is the goal of the model, $\mathbf{d}^S$ and $\mathbf{d}^{CHD}$ are treated as absorbing states and no rewards are accrued after entering these states.

Action Space: Initiating a cholesterol or blood pressure lowering medication is assumed to be an irreversible decision, which is consistent with the clinical practice in which the intention is for the patient to remain on the medication permanently. For each medication j, at each decision epoch, we either initiate this medication (I_j) or wait at least one period to initiate the medication (W_j). Therefore, for a living state, the action space is represented by

$$A(\mathbf{s}) = A_1(\mathbf{s}) \times \ldots \times A_M(\mathbf{s}) \quad \forall \mathbf{s} \in S_{\mathcal{L}}$$

where M is the total number of medications considered and

$$A_j(\mathbf{s}) = \begin{cases} \{I_j, W_j\} & \text{if } s_{R+j} = 0 \text{ and } \mathbf{s} \in S_{\mathcal{L}}, \\ \{W_j\} & \text{if } s_{R+j} = 1 \text{ and } \mathbf{s} \in S_{\mathcal{L}} \end{cases}$$

This simply means that there is a choice of whether to start medication j or not, provided that the patient is not already on medication j. At every decision epoch t, that decision maker selects an action $a_t \in A(s_t)$ when the system is in state s_t.

Initiating a medication is assumed to have a proportional change on each metabolic factor. Cholesterol medications are designed to lower total cholesterol and raise HDL, while blood pressure medications lower systolic blood pressure. It is assumed that cholesterol medications have negligible effect on blood pressure and vice versa since there is no evidence to the contrary. The estimates of the effects of these drugs on the metabolic values were obtained from [54].

Transition Probabilities: There are four types of probabilities in this MDP: the probability of non-diabetes-related death, probability of a CHD event, probability of a stroke, and the transition probabilities among living states. The estimates of these probabilities come from a combination of sources, including the literature and longitudinal patient data, as described below.

At epoch $t \in \mathcal{T}$, a non-diabetes-related death occurs with fixed probability p_t^O for every state $\mathbf{s} \in S_{\mathcal{L}}$. The probability p_t^O depends only on a patient's age and demographic information and can be estimated from mortality tables such as [36]. Note that we assume that p_t^O is independent of the risk factors for type 2 diabetes. If the patient is in state $\mathbf{s} \in S_{\mathcal{L}}$, a CHD or stroke event occurs with probability $p_t^C(\mathbf{s}, a_t)$ and $p_t^S(\mathbf{s}, a_t)$, respectively. These probabilities depend on the patient's age, metabolic state, current and initiated medications, as well as other attributes that affect risk such as race and gender. Estimates of these values can be obtained from risk models such as the *Framingham* model [28–30], the *UKPDS* model [33, 34], and the *ACC/AHA Pooled ASCVD risk calculator* [35]. These models fit risk equations to observational data for large cohorts of patients followed over many years to predict the probability of having an event within a certain time frame. Some models take the length of the time frame as an input to the equation, which gives an easy way to calculate the probability that corresponds to the time between epochs of the model. However, some models only give 10-year probabilities which must be adjusted to a 1-year probability to be used as an input to the MDP model. Reference [55] provides a discussion of converting the time-interval of transition probabilities to an adverse event or death state under the assumption that the rate of these events is constant. This assumption likely leads to some degree of over-estimation of the yearly transition probability, since the model suggests that as a patient ages, they are more likely to have an adverse event.

If the patient was in state $\mathbf{s} \in S_{\mathcal{L}}$ and did not enter an absorbing state, she will transition probabilistically among the living states, entering state $\mathbf{s}' \in S_{\mathcal{L}}$ with probability $p(\mathbf{s}'|\mathbf{s})$, which is given by

$$p(\mathbf{s}'|\mathbf{s}) = \left(\Pi_{r=1}^{R} p(s_r'|s_r)\right)\left(\Pi_{m=1}^{M} \mathbb{1}(s_{R+m}'|s_{R+m}, a_t)\right) \quad \forall \mathbf{s}, \mathbf{s}' \in S_{\mathcal{L}} \tag{6.1}$$

The first product in (6.1) indicates the probability of having the metabolic levels of state $\mathbf{s}'$ given the patient had the metabolic levels of state $\mathbf{s}$. This model assumes that HDL, total cholesterol, and blood pressure progress independently so that the

transition probability of all metabolic factors is simply the product of the transition probabilities within each metabolic factor. For a given metabolic factor, one can estimate the transition probabilities from a longitudinal patient data set (please see [42] for a detailed description of an estimation procedure). After segmenting the continuous values of the factor into discrete groups, one can count the total number of transitions from each group to every other group for the metabolic factor of interest. Dividing through by the total number of transitions out of the given group gives the transition probability. The model used in [54] estimated transition probabilities from observational data on 663 diabetes patients from the Mayo Electronic Records and Diabetes Electronic Management System, which is a medical record containing detailed results of laboratory results such as blood pressure, cholesterol, and HbA1c for diabetes patients at the Mayo Clinic. Note that relaxing the independence assumption of the progression of the metabolic factors would decrease the number of observed samples and therefore the method described above would not be desirable due to large sampling error. This is what motivates the independences assumption, which is supported by relatively low correlation between these risk factors.

In (6.1), the product of the indicator functions, $\mathbb{1}\{s'_{R+m}|s_{R+m}, a_t\}$ is used to distinguish between feasible transitions where $\mathbb{1}\{s'_{R+m}|s_{R+m}, a_t\} = 1$ if the transition from the medications used in state $\mathbf{s}$ to the medications used in state $\mathbf{s}'$ is valid given the actions taken in time t and 0 otherwise. For example, if a patient in state $\mathbf{s}$ was not on statins and the decision maker did not prescribe statins, then a transition to state $\mathbf{s}'$ in which statins are used is not possible. Since this is not a valid transition, the transition probability will be 0.

The complete set of transition probabilities are summarized in the following equation:

$$p_t(\mathbf{j}|\mathbf{s}, a_t) = \begin{cases} [1 - p_t^S(\mathbf{s}, a_t) - p_t^C(\mathbf{s}, a_t) - p_t^0] \cdot p(\mathbf{j}|\mathbf{s}) & \text{if } \mathbf{s},\ \mathbf{j} \in S_{\mathcal{L}} \\ p_t^S(\mathbf{s}, a_t) & \text{if } \mathbf{j} = \mathbf{d}^S \text{ and } \mathbf{s} \in S_{\mathcal{L}} \\ p_t^C(\mathbf{s}, a_t) & \text{if } \mathbf{j} = \mathbf{d}^C \text{ and } \mathbf{s} \in S_{\mathcal{L}} \\ p_t^O & \text{if } \mathbf{j} = \mathbf{d}^O \text{ and } \mathbf{s} \in S_{\mathcal{L}} \\ 1 & \text{if } \mathbf{s} = \mathbf{j} \in S_{\mathcal{D}} \\ 0 & \text{otherwise} \end{cases}$$

Rewards: As mentioned above, this model has a bi-criteria objective of maximizing the life years before the first CHD event or stroke while minimizing the discounted medication costs. To balance these competing objectives, we weight a life year (LY) by the willingness to pay factor, β. At epoch t, if the patient is in a living state, one life year is accrued with to give a reward of $r^{a_t}(\mathbf{s}) = \beta$. The decision maker also incurs a cost $c^{a_t}(\mathbf{s})$ which is the total yearly cost of the current medications of the patient in state $\mathbf{s}$ as well as any medications initiated by the selected action a_t at epoch t. In other words, the patient continues to accumulate rewards until she incurs a cardiovascular event or dies from other causes.

Solution Method: For a patient in state $\mathbf{s}$ in epoch t, let $V_t(\mathbf{s})$ denote the patient's maximum total expected dollar reward prior to her first CHD or stroke event or death. The following optimality equations define the optimal value function $V_t^*(\mathbf{s})$ and the optimal action in each state based on the optimal value function:

$$V_t^*(\mathbf{s}) = \max_{a_t \in A(\mathbf{s})} \left\{ r_t^{a_t}(\mathbf{s}) - c_t^{a_t}(\mathbf{s}) + \alpha \sum_{\mathbf{j} \in S} p_t(\mathbf{j}|\mathbf{s}, a_t) V_{t+1}^*(\mathbf{j}) \right\} \tag{6.2}$$

and

$$a_t^*(\mathbf{s}) \in \arg\max_{a_t \in A(\mathbf{s})} \left\{ r_t^{a_t}(\mathbf{s}) - c_t^{a_t}(\mathbf{s}) + \alpha \sum_{\mathbf{j} \in S} p_t(\mathbf{j}|\mathbf{s}, a_t) V_{t+1}^*(\mathbf{j}) \right\} \tag{6.3}$$

where $\alpha \in [0, 1)$ is the discount factor corresponding to the length between epochs, which is commonly set to 0.97 in health economic evaluations involving monetary costs and QALYs (see Chap. 7 of [56] for justification). $V_{N+1}^*(\mathbf{s})$ is assumed to be the expected discounted dollar reward accrued from period $N+1$ if the patient were to remain on the same medications given by state $\mathbf{s}$. Using $V_{N+1}^*(\mathbf{s})$ as a boundary condition, backward induction can be used to solve the MDP for the optimal decisions for each state and epoch. First, evaluate (6.2) at $t = N$ and proceed backwards until $t = 1$. The actions $a_t^*(\mathbf{s})$ that define the optimal policy are found by solving (6.3). Then, one can compare the optimal value function V_1^* to the value function V_1^π for any given policy π, which is of special interest when π is a common guideline used for cholesterol and blood pressure management.

6.4.2 Results: Comparison of Optimal Policies Versus Published Guidelines

In this section, we compare results for MDP-based policies with published treatment guidelines. In the United States, the guidelines for treatment of blood pressure and cholesterol are published by two independent committees. The Joint National Committee (JNC) is responsible for the American blood pressure guideline, while the Adult Treatment Panel (ATP) is responsible for the cholesterol guidelines [57, 58]. These guidelines have historically been "one size fits all" for diabetes patients and have not taken into account the individual risk profile of a patient. The action space of the model is consistent with the medications that these panels recommend. In this model, we consider statins and fibrates for cholesterol medications, and we consider the following blood pressure medications: thiazides, ACE-inhibitors, beta-blockers, and calcium-channel blockers.

The model in [54] used retrospective validation by comparing the results of the MDP with the outcomes of the patient cohort in the Framingham Heart Study

(FHS) [59]. The different outcomes are shown in Table 6.1. Most of the FHS diabetes patients were diagnosed after age 40 and so these patients provide a lower bound for the outcomes of patients diagnosed at age 40. The overall patient population of the FHS likely provide an upper bound on the outcomes of diabetic patients.

MDP model/Patient cohort	LYs before first event (after age 50)
FHS: diabetes patients	14.2 (12.3–16.1)
FHS: overall	21.2 (20.5–22.0)
Mason et al. [54], MDP: no treatment	18.9
Mason et al. [54], MDP: U.S. guideline	21.2

Table 6.1: Comparison of the expected LYs until the first event after age 50 from the MDP model presented with the model presented in [54] and the Framingham Heart Study (FHS). Confidence intervals are shown for the FHS

Differences between the FHS and this model could be due to imperfect adherence to guidelines, all other cause mortality, and differences in the underlying risk of the patient population. For example, the risk associated with heart disease and stroke has decreased significantly since the start of the Framingham study in 1948. Differences between the results we present below and those in the earlier model [54] differ because we have updated the model with data, such as all other cause mortality, that has been released since the publication of [54].

Figure 6.2 shows the optimal trade-off curve between the expected life years before the first event and the expected discounted medication costs. To obtain each curve, first we specified a risk model to estimate p_t^S and p_t^C. Then, we solved the corresponding MDP with different values of the willingness to pay factor, β. The labeled points on the vertical axis correspond to a β value of \$0/LY and the optimal policy is to never initiate treatment because there is no weight on the expected life years in this case. As the value of β increases, more medications tend to be initiated leading to increases in life years.

The U.S. guidelines are also shown on the graph. At the time of publication of [54], JNC 7 [60] and ATP III [61] were the guidelines in the United States. We used policy evaluation to determine how well these guidelines performed. Under each risk model assumption, the optimal policy can increase the expected time until the first event for the same medication cost used in the U.S. guidelines. Alternatively, the same life years until the first event that are achieved using these guidelines could be achieved at much lower cost with the optimal policy. See [54, 62] for additional results and comparison to international guidelines.

Figure 6.3 shows the optimal initiation of statins under different assumptions of the underlying risk model. The different risk models are functions of the patient's age, systolic blood pressure, HDL, and total cholesterol. The structure of these functions affects the optimal decisions associated with state.

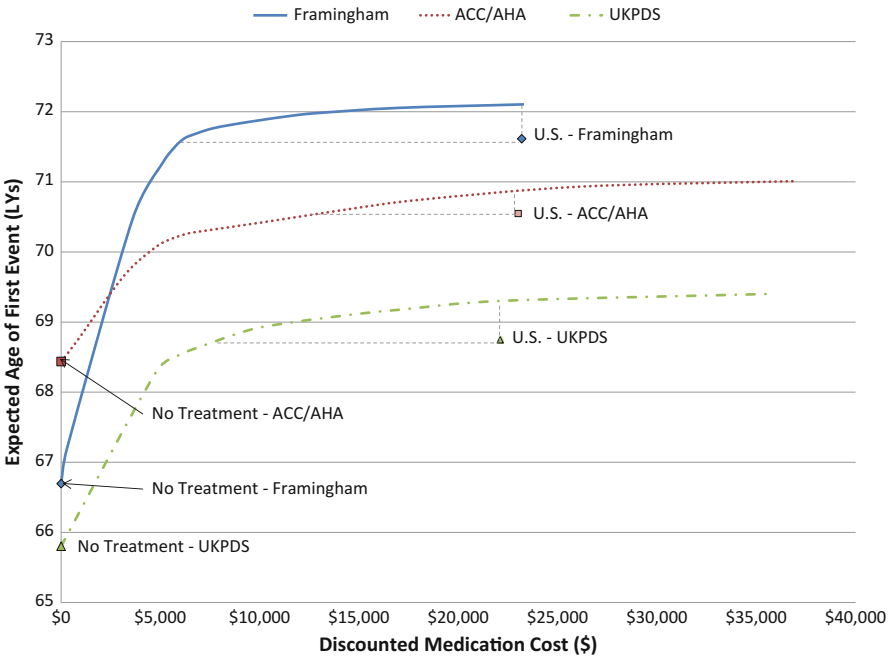


Fig. 6.2: Comparison of the expected life years until first event and discounted medication costs for optimal treatment policies and U.S. guidelines under different risk model assumptions

Figure 6.2 shows that coordinating the treatment of blood pressure and cholesterol could be beneficial for patients with type 2 diabetes under each of the three risk model assumptions. Because the underlying risk of complications is a function of both cholesterol and blood pressure, treating each risk factor separately, as recommended by the U.S. guidelines, could lead to higher cost and lower age of a first complication. This is supported by the outcomes of the U.S. guidelines which give high expected LYs and high discounted medication costs. This work shows that the optimal coordinated treatment of blood pressure and cholesterol depends on the underlying risk of the patient. However, as mentioned above, the risk models used to determine the probability of a complication often conflict with each other. For this reason, it would be beneficial to develop MDP methodology that provides policies that perform well despite disparities between the assumed risk model and the true underlying risk. Methods for achieving this goal remain an open research question.

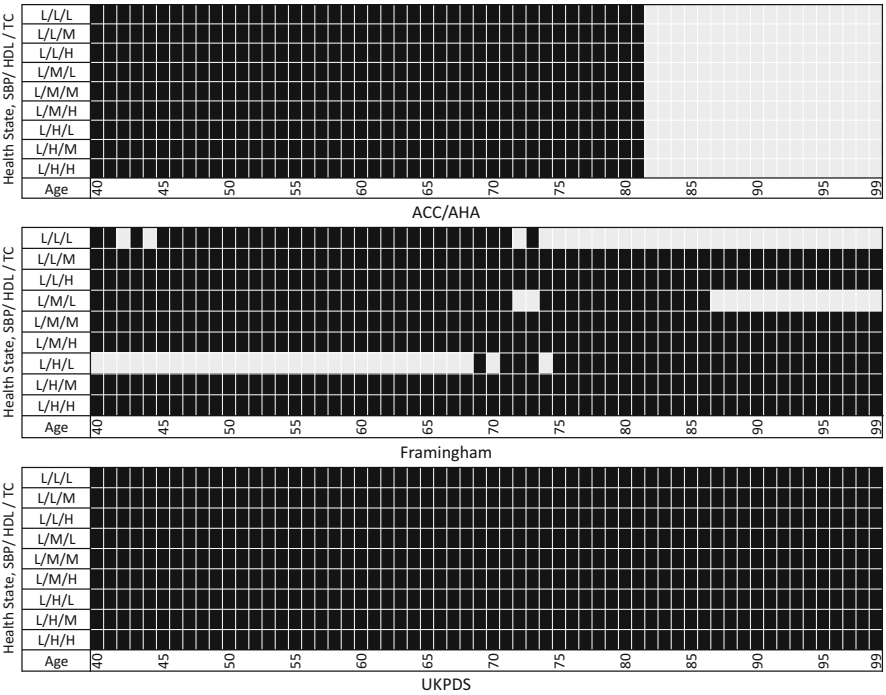


Fig. 6.3: A comparison of the optimal statin initiation actions under different risk model assumptions for β = \$50,000 per life year for selected blood pressure and cholesterol states. Black boxes indicate that the optimal decision is to initiate statins for this state and a white box indicates that the optimal decision is to wait to initiate statins. L/H/L is the healthiest state shown and L/L/H is the least healthy state shown

6.5 POMDP for Prostate Cancer Screening

Diagnosing chronic diseases is a challenge because most medical tests have some chance of *false positive* or *false negative* results. The former occurs when a test indicates a disease is present, when in fact it is not; the latter indicates a disease is not present, when in fact it is present. Successful diagnosis is critical to starting treatment early, and many chronic diseases, if detected early, have excellent outcomes. Prostate cancer is a good example. It is the most common cancer (excluding skin cancer) that affects men in many countries [21]. It is estimated that one in every seven U.S. men will be diagnosed with prostate cancer during his lifetime. Diagnosis is often based in part on a Prostate Specific Antigen (PSA) test that measures the amount of PSA in the blood. PSA varies from near zero to potentially high values (e.g. $>$20 ng/ml). Men with prostate cancer often have elevated levels of PSA, but this can also be caused by other non-cancerous conditions. A commonly used threshold for asserting that a biopsy is warranted is 4 ng/ml; however, this is subjec-

tive and it is has been observed to be associated with high false positive and false negative outcomes in the biopsy referral process. Figure 6.4 illustrates the imperfect nature of PSA testing using a receiver operating characteristic (ROC) curve. An ROC curve is generated by computing the true positive rate (also called *sensitivity* of the test) and one minus the false positive rate (also called *specificity* of the test) for various choices of the test threshold. Thus, the curve in Fig. 6.4 illustrates that, as the PSA threshold for biopsy increases, the true positive rate of biopsy referral based on the PSA test increases and the false positive rate decreases (a perfect test would have a true positive rate of one a false positive rate of zero). Different points on the curve correspond to different choices of the threshold at which to recommend biopsy.

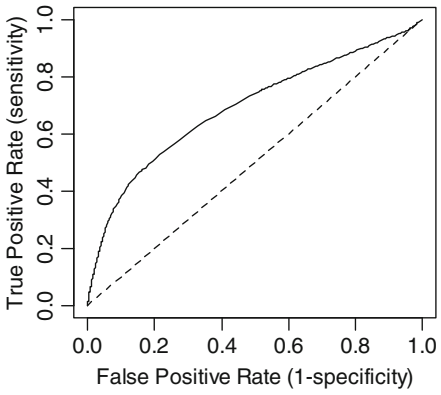


Fig. 6.4: Receiver operating characteristic (ROC) curve illustrating the imperfect nature of PSA tests for diagnosing prostate cancer

Given the invasive nature of prostate biopsies, the optimal threshold at which to recommend biopsy is debated. Moreover, the decision for when to biopsy must consider the fact that screening tests are often done multiple times over an individual's lifetime. An example of a screening process is illustrated in Fig. 6.5 where the patient receives routine PSA tests at regular intervals (often every year or every 2 years). If the PSA test result is over the biopsy threshold then the patient is typically referred for biopsy, and if the biopsy indicates cancer then the patient is referred for treatment. In practice, some clinicians consider the history of PSA test results for a patient, such as the rate of change with respect to time (often referred to as *PSA velocity*) because PSA is expected to increase with respect to tumor volume.

In this section, we present a POMDP model that uses an alternative approach for making screening decisions based on a patient's PSA history. In the model formulation that follows, Bayesian updating is used to estimate the probability that a patient has prostate cancer based on the complete history of PSA results. These probabilities are in turn used to decide when to perform a PSA test, and when to perform a

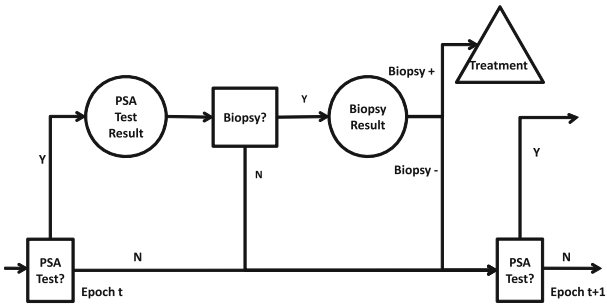


Fig. 6.5: Illustration of the typical stages of prostate cancer screening and treatment including PSA screening, biopsy, and treatment

biopsy. The model and the summary results we present are based on work presented in [63, 64] which together provide a complete description of the POMDP model, theoretical analysis of properties of the optimal policies, and a more complete description of the model parameters, results, and conclusions that can be drawn from the model.

6.5.1 POMDP Model Formulation

In the POMDP model, patients progress through (unobservable) prostate cancer states and (observable) PSA states. PSA states are treated as discrete, based on clinically relevant intervals, and estimated using a large observational data set. We assume that decision epochs occur annually, and the patient's PSA is measured at each epoch and a decision is made about whether to refer the patient for biopsy or defer the decision until the next epoch. Similar to the MDP model of the previous section, the POMDP model is also bi-criteria with the objective of maximizing a measure of life span minus the cost of screening and treatment for prostate cancer. To balance these competing objectives, we use a willingness to pay factor, β. In contrast to the application in Sect. 6.4, in this model QALYs are estimated by decrementing a normal life year due to the occurrence of biopsy (which is painful and has some, though low, probability of serious complications due to infection), side effects of treatment, and long term complications resulting from treatment. Costs are based on the cost of PSA tests, biopsies, and subsequent treatment. If a patient receives a positive biopsy result, he is assumed to be treated by prostatectomy (surgical removal of the prostate) which is a common form of treatment. If a patient

receives a negative biopsy result, then screening discontinues, an assumption that is motivated by the fact that most men have at most one biopsy in their lifetime unless other symptoms arise warranting additional biopsies. Following is a mathematical description of the model.

Decision Epochs: PSA screening is performed annually, typically starting at age 40, and thus the set of decision epochs is $\mathcal{T} \in \{40, 41, 42, \cdots, N\}$, where N corresponds to a liberal upper bound on when screening is discontinued due to the risk of treatment being greater than the benefits.

Time Horizon: This is a finite horizon model. The rewards accrued for patients that live beyond the last decision epoch N are estimated based on expected future survival given that the patient is not screened beyond epoch N ($N = 95$ in our numerical results).

State Space: At each decision epoch, a patient is in one of several health states including no cancer (NC), prostate cancer present but not detected (C), organ confined cancer detected (OC), extraprostatic cancer detected (EP), lymph node-positive cancer detected (LN), metastasis detected (M), and death from prostate cancer and all other causes (D). The states NC and C are not directly observable, but the other health states are assumed to be completely observable. The possible transitions among states are illustrated in Fig. 6.6. Note that state T in part (c) of Fig. 6.6 is an aggregate of treatment states OC, EP, and LN. This aggregation is possible since there are no actions in these states.

Action Space: The action at epoch t, $a_t \in \{B, DB, DP\}$, denotes the decision to perform a biopsy (B), defer biopsy and obtain a new PSA test result in epoch $t+1$ (DB), or defer the biopsy decision and PSA testing in decision epoch $t+1$ (DP). Combinations of these three actions over the decision horizon determine the PSA test and biopsy schedule. For instance, $a_{40} = DB$, $a_{41} = DP$, $a_{42} = DB$ and $a_{43} = B$ imply PSA testing at age 41 and 43, and followed by biopsy at age 43. Note that decisions are made sequentially and in this model decisions are based on the probability of prostate cancer at each decision epoch.

Observations: At each decision epoch, the patient is observed to be in one of a discrete set of observable PSA states based on clinically relevant ranges of PSA, non-metastatic cancer detected and treated (T), metastasis (M), or death (D). These observable states are indexed by $o_t \in O = \{1, 2, 3, \ldots, m, T, M, D\}$, where the first m states correspond to PSA states for patients either in state C or state NC. Note that the exact state, C or NC, cannot be known with uncertainty in this POMDP framework. The *observations* are a unique aspect of POMDP models that arise in the context of imperfect observability, as opposed to the completely observable context of Sect. 6.4.

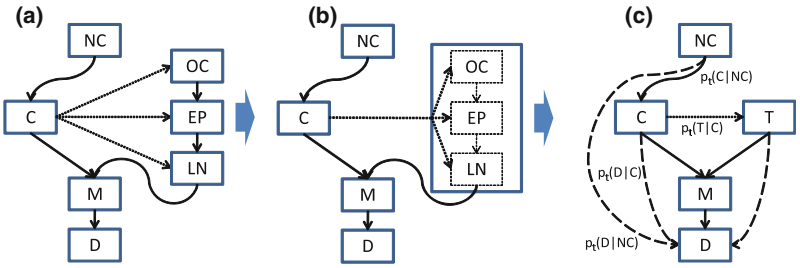


Fig. 6.6: POMDP model simplification: aggregating the three non-metastatic prostate cancer stages after detection into a single core state T. *Solid lines* denote the transitions related to prostate cancer; dotted lines denote the action of biopsy and subsequent treatment; *dashed lines* in (**c**) denote death from other causes (for simplicity these are omitted from (**a**) and (**b**)). (**a**) Prostate cancer developing flows. (**b**) State aggregation. (**c**) Core state transition

Transition Probabilities: The transition probability $p_t(s_{t+1}|s_t, a_t)$ denotes the core state transition probability from health state s_t to s_{t+1} at epoch t given action a_t. These represent the probability of a change in the patient's health status from one decision epoch to the next. By the nature of partially observable problems, such data is often difficult or impossible to estimate exactly. In the context of prostate cancer these estimates can be obtained using autopsy studies, in which all fatalities within a given region, regardless of cause of death, are investigated to determine the presence and extent of prostate cancer [65]. This provides estimates of the true incidence of disease that are not biased by the fact that diseases like prostate cancer may be latent for an extended period of time before diagnosis.

Information Matrix: A unique part of POMDP models, compared to MDP models, is the set of conditional probabilities that relate the underlying core states (e.g. *C* or *NC*) to the observations (e.g. PSA states). We let $u_t(o_t|s_t)$ denote the probability of observing $o_t \in O$ given health state $s_t \in S$. Collectively, these transition probabilities define the elements of the *information matrix*, which we denote by U_t. The estimation of these probabilities requires data that can link the observations to the cores states. Often this set of model parameters is the most difficult to estimate, because problems that are ideally modeled as partially observable are naturally ones in which limited data is available for the underlying core state of the system. Estimation of the information matrix is often made possible by a systematic randomized trial that evaluates the presence of disease independent of whether a patient has symptoms. In the case of prostate cancer, the *Prostate Cancer Prevention Trial (PCPT)* [66] had a protocol in which all men were biopsied independent of their PSA level. Based on data from this trial [67] fit a statistical model that can be used to estimated the probability a man has a given PSA level conditional on whether or not they are in state *C* or *NC*.

Belief States: The belief vector is a vector with elements each corresponding to one of the core states. In this model, each element corresponds to the probability the patient is in the corresponding core state. We denote the vector by $\mathbf{b}_t = (\mathbf{b}_t(NC), \mathbf{b}_t(C), \mathbf{b}_t(T), \mathbf{b}_t(M), \mathbf{b}_t(D))$, where $\mathbf{b}_t \in B \equiv \{\mathbf{b}_t \in \Re^5 \mid \sum_{i \in S} \mathbf{b}_t(i) = 1, \mathbf{b}_t(i) \geq 0, i \in S\}$. The optimal policy maps the belief states to the action space.

Rewards: $r^{a_t}(s_t)$ is the reward of living for a year given the patient is in health state s_t and decision a_t. The expected reward of living for a year is the average over possible health states: $r^{a_t}(\mathbf{b}_t) = \sum_{s_t \in S} r^{a_t}(s_t)\mathbf{b}_t(s_t)$. In this model, the reward is the product of QALYs and a willingness to pay factor minus the cost of a PSA test, biopsy or treatment, depending on the action a_t. The terminal reward at the end of the horizon, at period N is denoted $r_N(\mathbf{b}_t)$.

The overall objective of the model is to determine the optimal screening policy that maximizes the product of willingness to pay and QALYs minus the costs of screening and treatment over the patient's lifetime. The optimal value function and the corresponding optimal action for the model can be written as follows:

$$V_t^*(\mathbf{b}_t) = \max_{a_t \in \{B,DB,DP\}} \left\{ r^{a_t}(\mathbf{b}_t) + \alpha \sum_{o_{t+1} \in O} V_{t+1}^*(\mathbf{b}_{t+1}) p_t(o_{t+1}|\mathbf{b}_t, a_t) \right\}, \forall \mathbf{b}_t \in B, \tag{6.4}$$

and the boundary condition at the end of horizon is $V_N(\mathbf{b}_t) = r_N(\mathbf{b}_t), \forall \mathbf{b}_t \in B$. The optimal decision at epoch t in belief state $\mathbf{b}_t$ is

$$a_t^*(\mathbf{b}_t) \in \arg\max_{a_t \in \{B,DB,DP\}} \left\{ r^{a_t}(\mathbf{b}_t) + \alpha \sum_{o_{t+1} \in O} V_{t+1}^*(\mathbf{b}_{t+1}) p_t(o_{t+1}|\mathbf{b}_t, a_t) \right\}, \forall \mathbf{b}_t \in B,$$

where

$$p_t(o_{t+1}|\mathbf{b}_t, a_t) = \sum_{s_{t+1} \in S} u_{t+1}(o_{t+1}|s_{t+1}) \sum_{s_t \in S} p_t(s_{t+1}|s_t, a_t)\mathbf{b}_t(s_t),$$

and $\alpha \in [0,1)$ is the previously defined discount factor. Bayesian updating is used to revise the patient's belief state over time as PSA observations are obtained. Bayesian updates are defined by the following transformation of the belief state:

$$\mathbf{b}_{t+1}(s_{t+1}) = \frac{u_{t+1}(o_{t+1}|s_{t+1}) \sum_{s_t \in S} p_t(s_{t+1}|s_t, a_t)\mathbf{b}_t(s_t)}{\sum_{s_{t+1} \in S} u_{t+1}(o_{t+1}|s_{t+1}) \sum_{s_t \in S} p_t(s_{t+1}|s_t, a_t)\mathbf{b}_t(s_t)}, \tag{6.5}$$

where $\mathbf{b}_{t+1}(s_{t+1})$, the component of the belief vector, $\mathbf{b}_{t+1}$, is a function of o_{t+1}, a_t, and $\mathbf{b}_t$ (for simplicity, this dependence is omitted). Thus (6.5) updates the belief

state of a patient based on the prior belief state and his most recent observed PSA interval. The sequence of probabilities $\{\mathbf{b}_t, t = 1, \cdots, \infty\}$ has been shown to follow a Markov process [26], and therefore (6.4) defines a continuous state MDP.

6.5.2 Results: Optimal Belief-Based Screening Policy

In this section, we present examples based on the above POMDP model (complete details about model parameter estimates and numerical results can be found in [64]). The data used for parameter estimation in the model consisted of 11,872 patients from Olmsted County, Minnesota. It includes PSA values, biopsy information (if any), diagnosis information (if any), and the corresponding ages for patients recorded from 1983 through 2005. This regional data set includes all patients in Olmsted County irrespective of their prostate cancer risk. Prostate cancer probabilities conditional on PSA level were estimated from this data set to obtain the information matrix, U_t. In the results we present, we assume patients detected with prostate cancer were treated by radical prostatectomy. To estimate the annual transition probability from the treatment state, T, to the metastatic cancer state, M, a weighted average of the metastasis rate of three non-metastatic prostate cancer stages based on the Mayo Clinic Radical Prostatectomy Registry (MCRPR) were used. The disease specific transition probability from C to M was based on the metastasis rates reported by Catalona et al. [68]. The transition probability from state NC to state C was based on the prostate cancer incidence rate estimated from an autopsy review study reported in [69] that provides estimates of prostate cancer prevalence in the general population in 10-year age intervals. The transition probability from all non-cancer states to state D is age specific and was based on the general mortality rate from the National Vital Statistics Reports [70] minus the prostate cancer mortality rate from the [71]. Note that because the National Cancer Institute reports a single prostate cancer incidence rate for ages greater than 95 and the National Vital Statistics Reports [70] reports a single all cause mortality rate for ages greater than 95, we assume transition probabilities were fixed after the age of 95, i.e., $N = 95$ in the numerical experiments. The biopsy detection rate was 0.8 based on a study by Haas et al. [65]. To estimate the reward function we assumed an annual reward of 1 for each epoch minus disutilities for biopsy and treatment. Since no estimates of disutility exist yet for prostate biopsy, an estimate based on a bladder cancer study for the occurrence of surveillance cystoscopy [72] was used. We assumed patients treated by prostatectomy experience disutility due to side effects as reported in [73]. It is well known that POMDPs can be converted into an equivalent completely observable MDP on the continuous belief states $\mathbf{b}_t$ [43]. Even so, as noted earlier, POMDP models are typically much more computationally challenging to solve than completely observable MDPs, owing to the continuous nature of the belief state space. However, due to the low dimensionality of the belief state instances of this POMDP, it can be solved exactly using *incremental pruning* [44]. Incremental pruning is an exact solution method for POMDPs that builds upon early work of Monahan [74]. Monohan's approach enumerates the complete set of supporting hyperplanes (referred to

as α-vectors) that define the optimal value function for the POMDP. Incremental pruning attempts to reduce the computational effort required by decomposing the set of alpha vectors and pruning unnecessary (dominated vectors). See [75] for a review of POMDP properties and methods including incremental pruning.

The model was validated using a combination of expert opinion, based on feedback from practicing urologists, and comparison of the model results to independent studies. For the latter validation, the POMDP model was used to estimate mean lifespan, proportion of men diagnosed with prostate cancer, and prostate cancer mortality. These results were compared to published estimates from the CDC mortality tables and from longitudinal studies of diagnosis and mortality rates. See Chap. 3 of [76] for complete details of the validation of the model including comparison of the model results to those of a randomized trial.

Figure 6.7 illustrates the optimal prostate cancer screening policy based on the validated POMDP. Two examples are presented. In the first, only quality-adjusted life span is considered, and the costs of screening and treatment are not part of the reward function; this can be viewed as the *patient perspective*. In the second example, the reward function defines QALYs, weighted using a societal willingness to pay of $\beta = \$50{,}000/\text{QALY}$ [38] minus the cost of screening and treatment; as mentioned earlier, this can be viewed as the *societal perspective* since it weights the benefits of additional quality-adjusted lifespan against the cost of screening and treatment. The belief threshold between the three decisions is illustrated by the lines in the figure. From the figure, it can be observed that the optimal policy is control-limit type, a property that can be proven to hold under certain conditions for this POMDP [63]. A stopping time for screening occurs when the threshold for biopsy in the figures reaches 1.0. It is notable that there is a stopping time for screening at age 76 for the patient perspective. For the case of the societal perspective the stopping age is 71. The 5 year difference can be attributed to the cost of screening and treatment in the societal case. Finally, the policies in both examples demonstrates the optimal belief-based thresholds are age-dependent; as patients age, their probability of having prostate cancer must be higher in order for a biopsy referral to be optimal (action *B*). Moreover, the same is true for the decision to perform a PSA test (action *DB*). This is consistent with increasing all other cause mortality and the general consensus in the medical community that, due to the low mortality rate of prostate cancer, treatment becomes less beneficial as age increases.

6.6 Open Challenges in MDPs for Chronic Disease

While MDPs serve as a powerful tool for developing screening and treatment policies for chronic diseases, there exist open challenges in terms of formulating the MDPs and implementing the results from MDPs into clinical settings. We reflect on some of the challenges that were faced in the examples of the previous two sections:

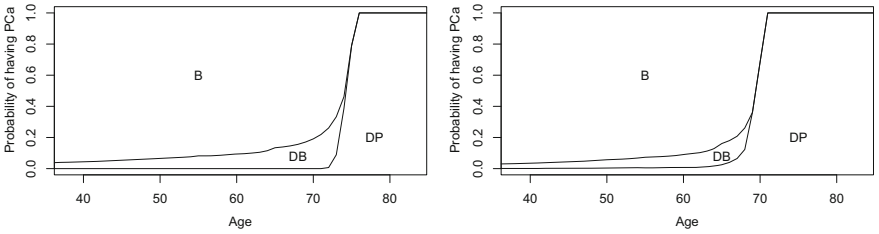


Fig. 6.7: Optimal prostate cancer screening policies from the patient perspective (*left*) and the societal perspective (*right*)

1. *Parameter Uncertainty*: Many estimates of the parameters used in chronic disease MDPs are subject to error. Transition probabilities among living states are usually estimated from observational data and therefore are subject to sampling error. Transitions to death states and adverse event states are estimated using risk models found in the literature, but usually there is no "gold standard" model. Further, estimates of disutilities due to medications are based on patient surveys and will vary patient-to-patient. As seen in Sect. 6.4, MDPs can be sensitive to changes in the model parameters, which is problematic when the model parameters cannot be known with certainty. For this reason, it will be important to develop MDP models that are *robust* in the sense that they perform well under a variety of assumptions of the model parameters, while not being overly conservative. The reader is referred to [77, 78] for more about robust dynamic programming.
2. *State Space Size and Transition Probability Estimates*: As discussed in Sect. 6.3, the continuously-valued metabolic risk factors are usually discretized to reduce the size of the state space. While a finer discretization of the state space might be more representative of the continuous-valued process, this will decrease the sample size of the transitions available for estimating transition probabilities. There is a natural trade-off between the fineness of the discretization of the state space and the error introduced in the transition probabilities due to sampling. Methods for determining the best discretization of the continuous state-space would reduce this barrier to formulating MDPs.
3. *Adjusting the Time Frame of Event Probabilities*: Many risk models provide the probability of an event or death within a fixed time (e.g. 10 years). While this information is useful to clinicians, MDP formulation requires converting these long-term probabilities into transition probabilities between epochs. As mentioned in Sect. 6.4, these probabilities can be converted under the assumption that the rate of events is constant, but this may not be realistic in all cases. Determining a method for converting probabilities under different assumptions about the rate of events would improve the accuracy of MDP models that use these risk probabilities.

4. *Solution Methods for a Large-Scale MDPs*: Chronic disease MDPs grow especially large because of the need to incorporate some history-dependence into the state space. Additionally, future models may incorporate risk factors for multiple, coexisting conditions which will cause the state space to grow ever larger. Because MDPs are subject to the curse of dimensionality, these models can be computationally-intensive to solve exactly. To provide support to clinicians in real-time, optimal policies should be able to be solved for quickly. This will not be possible in many chronic disease models, in which case fast approximation algorithms that provide near-optimal solutions will be necessary.
5. *Implementation of Optimal Policies*: The goal of these MDPs is to guide screening and treatment decisions made by the clinician. This requires that optimal policies can be made easily understood to clinicians. However, if the optimal policies are complicated, this could hinder the ability of the clinician to use the MDP results. Therefore, methods for designing structured policies that are near-optimal could potentially improve the likelihood of the policy being implemented in practice.

Tackling these challenges could make MDPs an even more useful tool for guiding clinicians and policy-makers in treatment and screening decisions.

6.7 Conclusions

Screening and treatment decisions for chronic disease are complicated by the long time periods over which these decisions are made and the uncertainty in the progression of disease, effects of medication, and correctness of test results. Throughout this chapter, we discussed a number of challenges that arise when modeling these decisions using MDPs, such as parameter uncertainty and the rapid growth in the size of the state space. Thus, there are still opportunities to study new application areas and develop new methodology, such as robust and approximate dynamic programming methods, for solving models in this context. These challenges notwithstanding, MDPs have recently found important applications to chronic diseases because they provide an analytical framework to study the sequential and dynamic decisions of screening and treating these diseases that develop stochastically over time.

Acknowledgements This material is based in part on work supported by the National Science Foundation under grant numbers CMMI 1462060 (Brian T. Denton) and DGE 1256260 (Lauren N. Steimle). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. World Health Organization, The top 10 causes of death (2013), Available at: who. int/mediacentre/factsheets/fs310/en/. 2014.
2. M.L. Brandeau, F. Sainfort, W.P. Pierskalla, *Operations Research and Health Care* (Kluwer Academic Publishers, Boston, 2004)
3. B.T. Denton, *Handbook of Healthcare Operations Management: Methods and Applications* (Springer, New York, 2013)
4. G.S. Zaric, *Operations Research and Health Care Policy* (Springer, New York, 2013)
5. O. Alagoz, L.M. Maillart, A.J. Schaefer, M.S. Roberts, The optimal timing of living donor liver transplantation. Manag. Sci. **50**(10), 1420–1430 (2004)
6. O. Alagoz, C.L. Bryce, S.M. Shechter, A.J. Schaefer, C.-C.H. Chang, D.C. Angus, M.S. Roberts, Incorporating biological natural history in simulation models: empiric estimates of the progression of end-stage liver disease. Med. Decis. Mak. **25**, 620–632 (2005)
7. O. Alagoz, L.M. Maillart, A.J. Schaefer, M.S. Roberts, Which waiting lists should an end-stage liver disease patient join? Technical Report, University of Pittsburgh, 2006
8. O. Alagoz, L.M. Maillart, A.J. Schaefer, M.S. Roberts, Choosing among living-donor and cadaveric livers. Manag. Sci. **53**(11), 1702–1715 (2007)
9. D.L. Segev, S.E. Gentry, D.S. Warren, B. Reeb, R.A. Montgomery, Kidney paired donation and optimizing the use of liver donor organs. J. Am. Med. Assoc. **295**, 1655–1663 (2005)
10. S.A. Zenios, G.M. Chertow, L.M. Wein, Dynamic allocation of kidneys to candidates on the trasplant waiting list. Oper. Res. **48**(4), 549–569 (2000)
11. X. Su, S. Zenios, Patient choice in kidney allocation: the role of the queuing discipline. Manuf. Serv. Oper. Manag. **6**(4), 280–301 (2005)
12. L.M. Maillart, J.S. Ivy, D. Kathleen, S. Ransom, Assessing dynamic breast cancer screening policies. Oper. Res. **56**(6), 1411–1427 (2008)
13. J. Chhatwal, O. Alagoz, E.S. Burnside, Optimal breast biopsy decision-making based on mammographic features and demographic factors. Oper. Res. **58**(6), 1577–1591 (2010)
14. E.K. Lee, T. Fox, I. Crocker, Integer programming applied to intensity-modulated radiation therapy treatment planning. Ann. Oper. Res. **119**, 165–181 (2003)
15. A. Holder, Designing radiotherapy plans with elastic constraints and interior point methods. Health Care Manag. Sci. **6**, 5–16 (2003)
16. F. Preciado-Walters, R. Rardin, M. Langer, V. Thai, A coupled column generation, mixed integer approach to optimal planning of intensity modulated radiation therapy for cancer. Math. Program. **101**, 319–338 (2004)
17. E.K. Lee, R.J. Gallagher, D. Silvern, C. Wu, M. Zaider, Treatment planning for brachytherapy: an integer programming model, two computational approaches, and experiments with permanent prostate implant planning. Phys. Med. Biol. **44**, 145–165 (1999)

18. S.M. Shechter, M.D. Bailey, A.J. Schaefer, M.S. Roberts, The optimal time to initiate HIV therapy under ordered health states. Oper. Res. **56**(1), 20–33 (2008)
19. E.H. Kaplan, Probability models of needle exchange. Oper. Res. **43**(4), 558–569 (1995)
20. G. Zaric, M.L. Brandeau, Optimal investment in a portfolio of HIV prevention programs. Med. Decis. Mak. **21**, 391–408 (2001)
21. R. Siegel, J. Ma, Z. Zou, A. Jemal, Cancer statistics, 2014. CA Cancer J. Clin. **64**(1), 9–29 (2014)
22. T. Ayer, O. Alagoz, N.K. Stout, A POMDP approach to personalize mammography screening decisions. Oper. Res. **60**(5), 1019–1034 (2012)
23. O. Alagoz, H. Hsu, A.J. Schaefer, M.S. Roberts, Markov decision processes: a tool for sequential decision making under uncertainty. Med. Decis. Mak. **30**(4), 474–483 (2010)
24. A.J. Schaefer, M.D. Bailey, S.M. Shechter, M.S. Roberts, Modeling medical treatment using Markov decision processes, in *Handbook of Operations Research/Management Science Applications in Health Care*, ed. by M. Brandeau, F. Sainfort, W. Pierskalla (Kluwer Academic, Dordrecht, 2004), pp. 597–616
25. E. Regnier, S.M. Shechter, State-space size considerations for disease-progression models. Stat. Med. **32**(22), 3862–3880 (2013)
26. G.E. Monohan, A survey of partially observable Markov decision processes: theory, models, and algorithms. Manag. Sci. **28**(1), 1–16 (1982)
27. W.S. Lovejoy, A survey of algorithmic methods for partially observed Markov decision processes. Ann. Oper. Res. **28**, 47–66 (1991)
28. K.A. Anderson, P.M. Odel, P.W.F. Wilson, W.B. Kannel, Cardiovascular disease risk profiles. Am. Heart J. **121**, 293–298 (1991)
29. P.A. Wolf, R.B. D'Agostino, A.J. Belanger, W.B. Kannel, Probability of stroke: a risk profile from the Framingham study. Stroke **22**(3), 312–318 (1991)
30. P.W.F Wilson, R.B. D?Agostino, D. Levy, A.M. Belanger, H. Silbershatz, W.B. Kannel, Prediction of coronary heart disease using risk factor categories. Circulation **97**(18), 1837–1847 (1998)
31. R.C. Turner, The uk prospective diabetes study - a review. Diabetes Care **21**, C35–C38 (1998)
32. T.M.E. Davis, H. Millns, I.M. Stratton, R.R. Holman, R.C. Turner, Risk factors for stroke in type 2 diabetes mellitus - United Kingdom Prospective Diabetes Study (UKPDS) 29. Arch. Intern. Med. **159**(10), 1097–1103 (1999)
33. R.J. Stevens, V. Kothari, A.I. Adler, I.M. Stratton, R.R. Holman, The UKPDS risk engine: a model for the risk of coronary heart disease in type ii diabetes (UKPDS 56). Clin. Sci. **101**(6), 671–679 (2001)
34. V. Kothari, R.J. Stevens, A.I. Adler, I.M. Stratton, S.E. Manley, H.A. Neil, R.R. Holman et al., UKPDS 60 risk of stroke in type 2 diabetes estimated by the UK Prospective Diabetes Study risk engine. Stroke **33**(7), 1776–1781 (2002)

35. D.C. Goff, D.M. Lloyd-Jones, G. Bennett, C.J. O'Donnell, S. Coady, J. Robinson, 2013 ACC/AHA guideline on the assessment of cardiovascular risk. J. Am. Coll. Cardiol. **129**, S49–S73 (2014)
36. S.L. Murphy, J. Xu, K.D. Kochanek, Deaths: final data for 2010. National Vital Statistics Reports: from the Centers for Disease Control and Prevention, National Center for Health Statistics. Natl. Vital Stat. Rep. **61**(4), 1–117 (2013)
37. D. Gold, M.R. Stevenson, D.G. Fryback, HALYS and QALYS and DALYS, oh my: similarities and differences in summary measures of population health. Annu. Rev. Public Health **23**, 115–134 (2002)
38. K.L. Rascati, The $64,000 question – what is a quality-adjusted life year worth? Clin. Ther. **28**(7), 1042–1043 (2006)
39. D.P. Bertsekas, J.N. Tsitsiklis, Neuro-dynamic programming: an overview, in *Proceedings of the 34th IEEE Conference on Decision and Control, 1995*, vol. 1 (IEEE, New York, 1995), pp. 560–564
40. W.B. Powell, *Approximate Dynamic Programming* (Wiley, Hoboken, 2007)
41. M.L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. (Wiley, Hoboken, 1994)
42. M. Kurt, B.T. Denton, A.J. Schaefer, N.D. Shah, S.A. Smith, The structure of optimal statin initiation policies for patients with type 2 diabetes. IIE Trans. Healthc. Eng. **1**, 49–65 (2011)
43. R.D. Smallwood, E.J. Sondik, The optimal control of partially observable Markov processes over a finite horizon. Oper. Res. **21**(5), 1071–1088 (1973)
44. A. Cassandra, M.L. Littman, N.L. Zhang, Incremental pruning: a simple, fast, exact method for partially observable Markov decision processes, in *Proceedings Thirteenth Annual Conference on Uncertainty in Artificial Intelligence, San Francisco, CA* (1997), pp. 54–61
45. A.R. Cassandra, L.P. Kaelbling, M.L. Littman. Acting optimally in partially observable stochastic domains. AAAI. Vol. 94 (1994). ftp://ftp.cs.brown.edu/pub/techreports/94/cs94-20.pdf
46. J.E. Eckles, Optimum replacement of stochastically failing systems, Ph.D. Dissertation, Dept. Eng.-Econ. Syst., Stanford University, Stanford (1966)
47. W.S. Lovejoy, Computationally feasible bounds for partially observed Markov decision processes. Oper. Res. **39**(1), 162–175 (1991)
48. Centers for Disease Control and Prevention, National Diabetes Statistics Report: Estimates of Diabetes and Its Burden in the United States, 2014. US Department of Health and Human Services, Atlanta, GA, 2014
49. V. Snow, M.D. Aronson, E.R. Hornbake, C. Mottur-Pilson, K.B. Weiss, Lipid control in the management of type 2 diabetes mellitus: a clinical practice guideline from the American College of Physicians. Ann. Intern. Med. **140**(8), 644–649 (2004)
50. D.G. Manuel, K. Kwong, P. Tanuseputro, J. Lim, C.A. Mustard, G.M. Anderson, S. Ardal, D.A. Alter, A. Laupacis, Effectiveness and efficiency of different guidelines on statin treatment for preventing deaths from coronary heart disease: modelling study. Br. Med. J. **332**(7555), 1419–1422 (2006)

51. P.N. Durrington, H. Prais, D. Bhatnagar, M. France, V. Crowley, J. Khan, J. Morgan, Indications for cholesterol-lowering medication: comparison of risk-assessment methods. Lancet, **353**(9149), 278–281 (1999)
52. B.T. Denton, M. Kurt, N.D. Shah, S.C. Bryant, S.A. Smith, Optimizing the start time of statin therapy for patients with diabetes. Med. Decis. Mak. **29**(3), 351–367 (2009)
53. J.E. Mason, D.A. England, B.T. Denton, S.A. Smith, M. Kurt, N.D. Shah, Optimizing statin treatment decisions for diabetes patients in the presence of uncertain future adherence. Med. Decis. Mak. **32**(1), 154–166 (2012)
54. J.E. Mason, B.T. Denton, N.D. Shah, S.A. Smith, Optimizing the simultaneous management of blood pressure and cholesterol for type 2 diabetes patients. Eur. J. Oper. Res. **233**(3), 727–738 (2014)
55. D.K. Miller, S.M. Homan, Determining transition probabilities confusion and suggestions. Med. Decis. Mak. **14**(1), 52–58 (1994)
56. M.R. Gold, J.E. Siegel, L.B. Russell, M.C. Weinstein, *Cost-Effectiveness in Health and Medicine* (Oxford University Press, New York, 1996)
57. P.A. James, S. Oparil, B.L. Carter, W.C. Cushman, C. Dennison-Himmelfarb, J. Handler, D.T. Lackland, M.L. LeFevre, T.D. MacKenzie, O. Ogedegbe et al., 2014 evidence-based guideline for the management of high blood pressure in adults: report from the panel members appointed to the eighth joint national committee (jnc 8). JAMA, **311**(5), 507–520 (2014)
58. N.J. Stone, J.G. Robinson, A.H. Lichtenstein, 2013 acc/aha guideline on the treatment of blood cholesterol to reduce atherosclerotic cardiovascular risk in adults: a report of the american college of cardiology/american heart association task force on practice guidelines. J. Am. Coll. Cardiol. **163**, 2889–2934 (2014); correction. J. Am. Coll. Cardiol. **63**(25), 3024–3025 (2014)
59. O.H. Franco, E.W. Steyerberg, F.B. Hu, J. Mackenbach, W. Nusselder, Associations of diabetes mellitus with total life expectancy and life expectancy with and without cardiovascular disease. Arch. Intern. Med. **167**(11), 1145–1151 (2007)
60. A.V. Chobanian, G.L. Bakris, H.R. Black, W.C. Cushman, L.A. Green, J.L. Izzo Jr., D.W. Jones, B.J. Materson, S. Oparil, J.T. Wright Jr. et al., The seventh report of the Joint National Committee on prevention, detection, evaluation, and treatment of high blood pressure: the JNC 7 report. JAMA **289**(19), 2560–2571 (2003)
61. National Cholesterol Education Program NCEP Expert Panel, Third report of the National Cholesterol Education program (NCEP) expert panel on detection, evaluation, and treatment of high blood cholesterol in adults (Adult Treatment Panel III) final report. Circulation **106**(25), 3143 (2002)
62. J. Shah, N.D. Mason, M. Kurt, B.T. Denton, A. Schaefer, V. Montori, S. Smith, Comparative effectiveness of guidelines for the management of hyperlipidemia and hypertension for type 2 diabetes patients. Plos One **6**(1), e16170 (2011)
63. J. Zhang, H. Balasubramanian, B.T. Denton, N. Shah, B. Inman, Optimization of prostate cancer screening decisions: a comparison of patient and societal perspectives. Med. Decis. Mak. **32**(2), 337–349 (2011). doi:10.1177/0272989X11416513

64. J. Zhang, B.T. Denton, H. Balasubramanian, N.D. Shah, B.A. Inman, Optimization of prostate biopsy referral decisions. Manuf. Serv. Oper. Manag. **14**(4), 529–547 (2012)
65. G.P. Haas, R.F. Delongchamps, V. Jones, V. Chandan, A.M. Seriod, A.J. Vickers, M. Jumbelic, G. Threatte, R. Korets, H. Lilja, G. De la Roza, Needle biopsies on autopsy prostates: sensitivity of cancer detection based on true prevalence. J. Natl. Cancer Inst. **99**, 1484–1849 (2007)
66. I.M. Thompson, D.P. Ankerst, C. Chi, P.J. Goodman, C.M. Tangen, M.S. Lucia, Z. Feng, H.L. Parnes, C.A. Coltman, Assessing prostate cancer risk: results from the prostate cancer prevention trial. J. Natl. Cancer Inst. **98**(8), 529–534 (2006)
67. R. Gulati, L. Inoue, J. Katcher, W. Hazelton, R. Etzioni, Calibrating disease progression models using population data: a critical precursor to policy development in cancer control. Biostatistics **11**(4), 707–719 (2010)
68. W.J. Catalona, P.T. Scardino, J.R. Beck, B.J. Miles, G.W. Chodak, R.A. Thisted, Conservative management of prostate cancer. N. Engl. J. Med. **330**(25), 1830–1832 (1994)
69. L. Bubendorf, A. Schöpfer, U. Wagner, G. Sauter, H. Moch, N. Willi, T.C. Gasser, M.J. Mihatsch, Metastatic patterns of prostate cancer: an autopsy study of 1,589 patients. Hum. Pathol. **31**(5), 578–583 (2000)
70. M. Heron, Deaths: leading causes for 2004. Natl. Vital Stat. Rep. **56**(5), 1–96 (2007)
71. National Cancer Institute, Surveillance Epidemiology and End Results (SEER). SEER Stat Fact Sheets, Cancer: Prostate (2009). http://seer.cancer.gov/statfacts/html/. Accessed May 2015
72. G.S. Kulkarni, S.M.H. Alibhai, A. Finelli, N.E. Fleshner, M.A.S. Jewett, S.R. Lopushinsky, A.M. Bayoumi, Cost-effectiveness analysis of immediate radical cystectomy versus intravesical bacillus Calmette-Guerin therapy for high-risk, high-grade (t1g3) bladder cancer. Cancer **115**(23), 5450–5459 (2009)
73. K.E. Bremner, C.A.K.Y. Chong, G. Tomlinson, S.M.H Alibhai, M.D Krahn, A review and meta-analysis of prostate cancer utilities. Med. Decis. Making **27**, 288–298 (2007)
74. G.E. Monohan, A survey of partially observable Markov decision processes: theory, mondels, and algorithms. Manag. Sci. **28**(1), 1–16 (1982)
75. L.P. Kaelbling, M.L. Littman, A.R. Cassandra, Planning and acting in partially observable stochastic domains. Artif. Intell. **101**(1), 99–134 (1998)
76. D. Underwood, Risk-based simulation optimization of PSA-based prostate cancer screening. Ph.D. Thesis, North Carolina State University, 2015
77. G.N. Iyengar, Robust dynamic programming. Math. Oper. Res. **30**(2), 257–280 (2005)
78. A. Nilim, L.E. Ghaoui, Robust control of Markov decision processes with uncertain transition matrices. Oper. Res. **55**(5), 780–798 (2005)

Chapter 7
Stratified Breast Cancer Follow-Up Using a Partially Observable MDP

J.W.M. Otten, A. Witteveen, I.M.H. Vliegen, S. Siesling, J.B. Timmer, and M.J. IJzerman

Abstract Frequency and duration of follow-up for patients with breast cancer is still under discussion. Current follow-up consists of annual mammography for the first five years after treatment and does not depend on the personal risk of developing a locoregional recurrence (LRR) or second primary tumor. Aim of this study is to gain insight in how to allocate resources for optimal and personal follow-up. We formulate a discrete-time Partially Observable Markov Decision Process (POMDP) with a finite horizon in which we aim to maximize the total expected number of quality-adjusted life years (QALYs). Transition probabilities were obtained from data from the Netherlands Cancer Registry (NCR). Twice a year the decision is made whether or not a mammography will be performed. Recurrent disease can be detected by both mammography or women themselves (self-detection). The optimal policies were determined for three risk categories based on differentiation of the primary tumor. Our results suggest a slightly more intensive follow-up for patients with a high risk and poorly differentiated tumor, and a less intensive schedule for the other risk groups.

J.W.M. Otten • J.B. Timmer
Department of Stochastic Operations Research, University of Twente, Enschede, The Netherlands

A. Witteveen (✉) • M.J. IJzerman
Department of Health Technology and Services Research, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands
e-mail: a.witteveen@utwente.nl

I.M.H. Vliegen
Department of Industrial Engineering and Business Information Systems, University of Twente, Enschede, The Netherlands

S. Siesling
Department of Health Technology and Services Research, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands

Department of Research, Comprehensive Cancer Organisation, Utrecht, The Netherlands

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_7

7.1 Introduction

After curative treatment for breast cancer, patients are followed clinically to detect locoregional recurrences (LRRs) in an early phase [1]. A LRR is defined as the reappearance of breast cancer on the same site as the primary tumour [2]. Currently, in the Netherlands, patients have annual follow-up for at least five years [3]. However, only a minority of the LRRs discovered are detected at a scheduled check-up [4]. Furthermore, due to an increasing incidence and survival rate, the number of patients currently in the follow-up phase increases and becomes a higher burden to health care. Even though risk factors are known, there is no differentiation in the follow-up policy for different categories of patients [5].

These observations together give rise to the question whether it is possible to improve the current follow-up policy, both from a patient's and health care perspective. Getting towards more personalized follow-up asks for specific modelling requirements, because of the sequential decisions and individual risks and effects [6]. Our aim is to develop a sequential decision process in which a decision maker, the patient or a physician, chooses at every decision epoch whether or not to have a check-up. We model this problem using a partially observable Markov decision process (POMDP), which is a generalization of a Markov decision process and allows us to model a sequential decision making process in which the information about the true state of the system is incomplete [7]. Because the true health state of a patient, i.e., whether a patient is disease-free or not, is only partially observable, a POMDP is ideally suited to this problem [8].

Ayer et al. [9] developed a POMDP model for a similar problem, a mammography decision model for screening for primary breast cancer. This model resembles our problem, although necessary adjustments need to be made. Firstly, because we apply the model to a different population, namely breast cancer patients in follow-up instead of a health screening population the need to incorporate self-detection differs. When applying the model to the whole population, self-detection is a less important factor to take into account than when applying it to breast cancer patients in the follow-up phase, since a large part of the LRR detections is via self-detection [4]. Secondly, because we model the decision process over a short horizon instead of over the whole lifetime of a patient, adjustments concerning final rewards need to be made. Ayvaci et al. [10] give an analysis of the same problem as Ayer et al. but under budgetary constraints, however they model it as a normal Markov decision process and thus neglecting the partial observability of the true health state. Zhang et al. [11] made a comparison of the patient and societal perspectives for a similar case, PSA screening policies for prostate cancer, via a POMDP approach.

Our contributions to this research are twofold. Firstly, we apply the POMDP approach to a problem for which it is not previously done and develop a model from which a personal testing schedule can be obtained, based on a patient's personal characteristics and test history. Secondly we discover that the outcomes of the POMDP model are quite sensitive to certain input parameters, mostly to the growth rate of a LRR and the life expectancy of a patient with breast cancer, so that better estimates of these parameters are necessary in order to apply this model in practice.

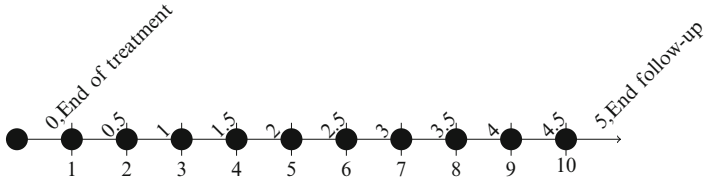
The remainder of this chapter is organized as follows. In Sect. 7.2 we present the POMDP model by which the problem is modelled and derive the optimality equations, from which the solution algorithm is deduced. In Sect. 7.3 we describe the model parameters. In Sect. 7.4 we present the results of the model and perform sensitivity analyses for some of the parameters. Finally we summarize the results and conclude in Sect. 7.5.

7.2 Model Formulation

The problem described is modelled by a discrete-time partially observable Markov decision process (POMDP) with a finite horizon of five years, in which the objective is to maximize the total expected number of quality-adjusted life years (QALYs).

Twice a year, a decision is made whether a patient should have a mammogram or should wait for another six months. The choice for biannual decisions is based on the wish to develop a personal decision model in which, ideally, the check-ups could be arranged at any moment in the follow-up phase. Since the problem is modeled by a discrete POMDP, biannual decisions are a first step in that direction. Because the true health state of a patient is not known, the decisions are made based on the present belief about the patient's health state. This belief may depend on several personal risk factors and the patient's test history. When the decision is made that the patient should have a mammogram and the result is positive or if a self-detection is made, it is followed by a biopsy. We assume that the biopsy after a positive mammogram or self-detection is perfect. When this perfect test is positive as well (i.e. the patient has cancer), we assume that treatment is started immediately and that the patient will leave the decision process by moving to one of the absorbing treatment states. Otherwise the decision process proceeds to the next decision epoch. When the mammogram is negative or the decision is to wait for another six months and no self-detection is made, the decision process also proceeds to the next decision epoch where the same decision has to be made. For our notation we follow Ayer et al. [9]. The complete model and the notation used is as follows:

• Decision epochs, $t = 1 \cdots T$, $T = 10$. Decisions are made twice every year and the decision process starts six months after treatment of the primary tumour finished, so $t = 5$ denotes 2.5 years after primary treatment (see timeline below). Let τ denote the time between two successive decision epochs, $\tau = 0.5$ year. The decision horizon is at $T = 10$ because the annual check-ups are stopped after five years (depending on the age of the patient the check-ups in the hospital after this are annual, biennial or stopped) [3].



• Core state space, $S = \{1,2,3,4,5,6,7,8\}$. With 1 for no (detectable) cancer, 2 for a LRR when early detection is still possible, 3 for a LLR when early detection is not possible any more and 4 for a second primary (SP) tumour (another incidence of breast cancer independent of the primary tumour). We distinguish different stages of a LRR due to the difference in expected remaining QALYs: early detection of a LRR yields a better prognosis [1]. The states 5, 6 and 7 stand for treatment for the various cancer states and finally 8 stands for death of the patient. Figure 7.1 shows how these different states are connected. The state s_t is the true health state of the patient at time t. This state of the patient is not directly observable when the patient is in state 1, 2, 3 or 4. The other states of the patient, the 'Treatment' states and the state 'Death', can be directly observed. We therefore call the states $\{1,2,3,4\}$ partially observable and denote this subset of the core state space as S^{PO}.

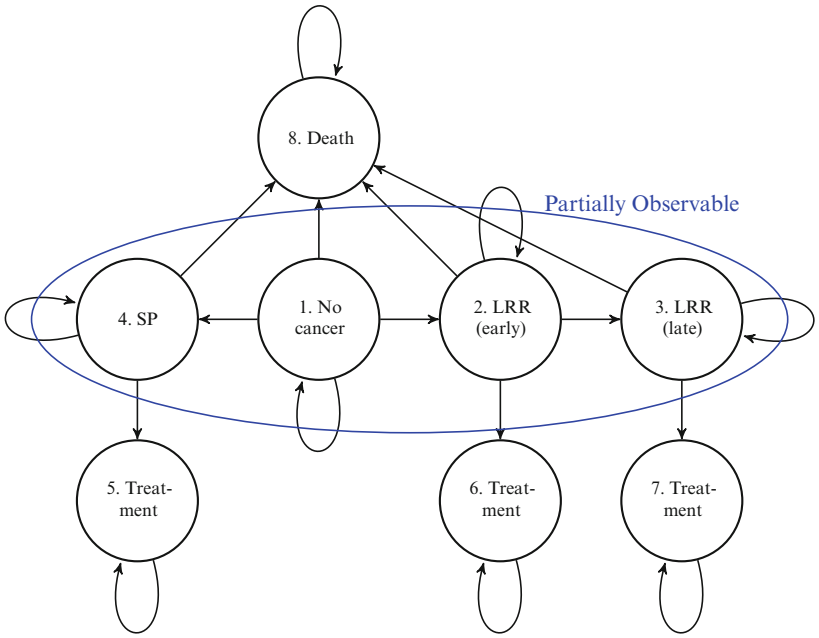


Fig. 7.1: State diagram of the underlying Markov process

• Information space, $\Pi(S)$, the space of all probability distributions over the core state space S. A vector $\pi \in \Pi(S)$ is called an information state and $\pi(s), s \in S$, denotes the probability that the true state is s.

• Belief space, $B(S^{PO})$, the space of all probability distributions over the partially observable states, S^{PO}. A vector $b \in B(S^{PO})$ is in fact a truncated version of the vector $\pi \in \Pi(S)$, because the probability that the true state $s = 5,6,7,8$ is either 0 or 1.

• Actions, A'_t, the set of actions a_t at time t. Here $A'_t = \{W, M\}$, where W stands for wait and M for mammography. The action set is only defined for $s \in S^{PO}$ since the process terminates whenever the patient dies or goes to a treatment state.

• Observation space, Θ_a, the set of possible observations when action a is selected. If $a_t = M$, the possible observations are a positive mammogram (M^+) or a negative mammogram (M^-). If $a_t = W$, the patient can either make a self-detection (SD^+) or no self-detection (SD^-). So we have $\Theta_M = \{M^+, M^-\}$ and $\Theta_W = \{SD^+, SD^-\}$.

• Observation probabilities. $K_t^a(o|s)$ is the probability of observing o at time t when decision a was taken while in state s. These probabilities are completely determined by the specificity (true negative rate) and the sensitivity (true positive rate). For example, $K_t^M(M^-|s=1)$ is the probability of having a negative mammogram when the true health state of the patient is $s = 1$ ('No cancer'), this is the specificity of the mammogram. We denote the specificity of mammography by $spec_t(M)$ and of self-detection by $spec_t(SD)$. Similarly, the sensitivity of mammography is denoted by $sens_t(s,M)$ and of self-detection $sens_t(s,SD)$. Note that, unlike specificity, the sensitivity of a test depends on the true health state of the patient because there are multiple cancer states. From these observations we can obtain the observation probabilities as follows:

$$
\begin{aligned}
K_t^M(M^-|s=1) &= spec_t(M) \\
K_t^M(M^+|s=1) &= 1 - spec_t(M) \\
K_t^W(SD^-|s=1) &= spec_t(SD) \\
K_t^W(SD^+|s=1) &= 1 - spec_t(SD) \\
K_t^M(M^+|s=x) &= sens_t(s,M) && x = 2,3,4 \\
K_t^M(M^-|s=x) &= 1 - sens_t(s,M) && x = 2,3,4 \\
K_t^W(SD^+|s=x) &= sens_t(s,SD) && x = 2,3,4 \\
K_t^W(SD^-|s=x) &= 1 - sens_t(s,SD) && x = 2,3,4
\end{aligned}
$$

• Core state transition probabilities. $P_t^{(a,o)}(s'|s)$ is the probability that the true health state of the patient at time $t+1$ is s' given it was s, action a was chosen and o was observed. The transition of the true health state between time t and $t+1$ does not depend on the action taken or the observation made at time t. Therefore $P_t^{(M,M^-)}(s'|s) = P_t^{(W,SD^-)}(s'|s) = P_t^{(W,SD^+)}(s'|s)$ for all $s', s \in S$ and $P_t^{(M,M^+)}(s'|s=0) = P_t^{(M,M^-)}(s'|s=0)$.

• Updated belief space. Let $\tau[b,a,o]$ denote the belief (i.e. the probability distribution over the partially observable states) at time $t+1$ when the belief about the patient's true health state at time t was b, action a was taken and observation o was made. In particular, $\tau[b,a,o](s) = P_{t+1}(s|b,a,o)$ for all states $s \in S^{PO}$.

The updated belief state is computed by:

$$\tau[b,a,o](s') = \begin{cases} \frac{\sum_{s\in S^{PO}} b(s)K_t^a(o|s)P_t^{a,o}(s'|s)}{\sum_{s\in S^{PO}} b(s)K_t^a(o|s)} & \text{if } a=W,\ o=SD^- \\ & \quad \text{or } a=M,\ o=M^-, \\ P_t^{W,SD^+}(s'|1) & \text{if } a=W,\ o=SD^+, \\ P_t^{M,M^+}(s'|1) & \text{if } a=M,\ o=M^+. \end{cases} \tag{7.1}$$

The complete derivation and proof can be found in [7].

• Rewards. The reward $r_t(s,a,o)$ is the expected number of QALYs between two decision epochs when the true health state of the patient is s, action a is taken and observation o was made. To factor in the probability that a patient dies between two decisions we use the half-cycle correction method [12]. The idea of this correction method is that if the patient dies between two decision epochs, it is assumed that half of the cycle length τ is accrued to the expected number of QALYs. From this, QALYs are subtracted for the disutility of a mammogram and a biopsy, when a patient should have one of these. Note that if the patient is in one of the cancer states ($s \in \{2,3,4\}$) and observes a positive mammogram or makes a self-detection, she is rewarded a lump-sum reward (LSR) of $R_t(s,a)$. So no QALYs are rewarded over the next decision epoch when a true positive mammogram or self-detection is observed, i.e. $r_t(s,M,M^+) = r_t(s,W,SD^+) = 0$, $s = 2,3,4$. The rewards in the treatment and death states are zero.

The expected reward between time t and $t+1$ when the true health state is s and the action chosen is a is denoted by $r_t(s,a) = \sum_{o\in\Theta_a} K_t^a(o|s) r_t(s,a,o)$.

Let $r_T(s)$ denote the total expected remaining QALYs at time T when the patient's true health state is s at time T.

Let p_d denote the probability that a patient dies between two decision epochs and dis_M, dis_B the disutility experienced when undergoing a mammogram and a biopsy, respectively. The rewards for $t = 1,\cdots,T-1$ are:

$$\begin{aligned} r_t(s,W,SD^-) &= p_d\cdot 0.5\tau + (1-p_d)\cdot\tau && s\in S^{PO} \\ r_t(1,W,SD^+) &= p_d\cdot 0.5\tau + (1-p_d)\cdot\tau - dis_B && \\ r_t(s,M,M^-) &= p_d\cdot 0.5\tau + (1-p_d)\cdot\tau - dis_M && s\in S^{PO} \\ r_t(1,M,M^+) &= p_d\cdot 0.5\tau + (1-p_d)\cdot\tau - dis_M - dis_B && \\ r_t(s,\cdot,\cdot) &= 0 && \text{otherwise} \end{aligned}$$

7.2.1 Optimality Equations

We want to derive optimality equations for the number of QALYs a patient can obtain in order to determine the optimal policy for a patient. Let $V_t^*(\pi)$ denote this quantity when her information state is $\pi \in \Pi(S)$ at time t. Likewise let $V_t^*(b)$ denote the same quantity when the patient's belief state is $b \in B(S^{PO})$. Because the process

terminates when in one of the treatment states or the death state, $V_t^*(\pi)$ can be expressed as:

$$V_t^*(\pi) = \begin{cases} R_t(2) & \pi = [0\,0\,0\,0\,1\,0\,0\,0], \\ R_t(3) & \pi = [0\,0\,0\,0\,0\,1\,0\,0], \\ R_t(4) & \pi = [0\,0\,0\,0\,0\,0\,1\,0], \\ V_t^*(b) & b = [b(1)\ b(2)\ b(3)] \neq [0\,0\,0], \\ 0 & \text{otherwise .} \end{cases} \tag{7.2}$$

Let $V_t^a(b)$ denote the maximum total expected QALYs a patient can obtain when at time t in belief state b and action a is chosen. Then $V_t^*(b)$ can be written as:

$$\begin{aligned} V_t^*(b) &= \max\{V_t^W(b), V_t^M(b)\} \qquad t = 1 \cdots T-1, \qquad \text{with} \\ V_t^W(b) &= \sum_{s \in S^{PO}} b(s)[K_t^W(SD^-|s)[r_t(s,W,SD^-) \\ &\quad + \sum_{u \in S} P_t^{(W,SD^-)}(u|s)V^*(\tau[b,W,SD^-])]] \\ &\quad + b(1)K_t^W(SD^+|1)[r_t(1,W,SD^+) \\ &\quad + \sum_{u \in S} P_t^{(W,SD^+)}(u|s)V^*(\tau[b,W,SD^+])] \\ &\quad + \sum_{s=2}^{4} b(s)K_t^W(SD^+|s)R_t(s,W) \qquad \text{and} \end{aligned}$$

$$\begin{aligned} V_t^M(b) &= \sum_{s \in S^{PO}} b(s)[K_t^M(M^-|s)[r_t(s,M,M^-) \\ &\quad + \sum_{u \in S} P_t^{(M,M^-)}(u|s)V^*(\tau[b,M,M^-])]] \\ &\quad + b(1)K_t^M(M^+|1)[r_t(1,M,M^+) \\ &\quad + \sum_{u \in S} P_t^{(M,M^+)}(u|s)V^*(\tau[b,M,M^+])] \\ &\quad + \sum_{s=2}^{4} b(s)K_t^M(M^+|s)R_t(s,M) \\ V_T^*(b) &= \sum_{s \in S^{PO}} b(s)r_T(s) \end{aligned}$$

The optimality equations can be simplified by moving the parts that do not depend on u outside the summations over u and by noting that $\sum_{u \in S^{PO}} P^{(a,o)}(u|s) = 1$.

$$
\begin{aligned}
V_t^*(b) = \max \Bigg\{ & \sum_{s \in S^{PO}} b(s)[K_t^W(SD^-|s)[r_t(s,W,SD^-) + V^*(\tau[b,W,SD^-])]] \\
& + b(1)K_t^W(SD^+|1)[r_t(1,W,SD^+) + V^*(\tau[b,W,SD^+])] \\
& + \sum_{s=2}^{4} b(s)K_t^W(SD^+|s)R_t(s,W), \\
& \sum_{s \in S^{PO}} b(s)[K_t^M(M^-|s)[r_t(s,M,M^-) + V^*(\tau[b,M,M^-])]] \\
& + b(1)K_t^M(M^+|1)[r_t(1,M,M^+) + V^*(\tau[b,M,M^+])] \\
& + \sum_{s=2}^{4} b(s)K_t^M(M^+|s)R_t(s,M) \Bigg\} \qquad t = 1 \cdots T-1 \\
V_T^*(b) = & \sum_{s \in S^{PO}} b(s) r_T(s)
\end{aligned} \tag{7.3}
$$

7.2.2 Alternative Representation of the Optimality Equations

The key idea in solving any type of Markov decision process is relating the optimal value function V^* at time t to V^* at time $t+1$. As derived in the previous section the value function of this particular problem is defined on the continuous space $B(S^{PO}) = \left\{ b \in \mathbb{R}^4 | \sum_{s \in S^{PO}} b(s) = 1, b(s) \geq 0 \right\}$. So for solving the optimal value function at time t one would need the optimal value function at time $t+1$ on a continuous space and therefore an infinite dimensional vector would be needed to store these value functions. Fortunately it can be proven that for a POMDP the optimal value function is piecewise linear and convex, and can therefore be represented as the maximum over a finite number of finite-dimensional vectors. This result is stated in the following theorem.

Theorem 7.1 (Smallwood and Sondik [7]). *The optimal value function $V_t^*(b)$ is piecewise linear and convex, and can thus be written as*

$$
V_t^*(b) = \max_k \sum_{s \in S^{PO}} b(s) \alpha_t^k(s), \tag{7.4}
$$

for some set of vectors $\alpha_t^k = [\alpha_t^k(s)]$, $k = 1, 2, \cdots$. The term α-vector is used to refer to such a vector.

The optimality equations can now be written in terms of the α-vectors. The proofs of the two propositions below are similar to results in Ayer et al. [9]

Proposition 7.1. *The following representation of the optimality equations is equivalent to the optimality equations given in (7.3).*

$$
\begin{aligned}
V_t^*(b) = \max \Bigg\{ & \sum_{s \in S^{PO}} b(s) K_t^W(SD^-|s) [r_t(s,W,SD^-) \\
& + \sum_{u \in S^{PO}} P_t^{(W,SD^-)}(u|s) \alpha_{t+1}^{i(b,W,SD^-)}(u) \\
& + b(1) K_t^W(SD^-|1) [r_t(1,W,SD^-) \\
& + \max_k \left\{ \sum_{u \in S^{PO}} P_t^{(W,SD^-)}(u|1) \alpha_{t+1}^k(u) \right\} \\
& + \sum_{s=2}^{4} b(s) K_t^W(SD^+|s) R_t(s,W), \qquad (W) \\
& \sum_{s \in S^{PO}} b(s) K_t^M(M^-|s) [r_t(s,M,M^-) \\
& + \sum_{u \in S^{PO}} P_t^{(M,M^-)}(u|s) \alpha_{t+1}^{i(b,M,M^-)}(u) \\
& + b(1) K_t^M(M^-|1) [r_t(1,M,M^-) \\
& + \max_k \left\{ \sum_{u \in S^{PO}} P_t^{(M,M^-)}(u|1) \alpha_{t+1}^k(u) \right\} \\
& + \sum_{s=2}^{4} b(s) K_t^M(M^+|s) R_t(s,M) \Bigg\} \qquad (M)
\end{aligned} \tag{7.5}
$$

Where

$$
i(b,a,o) = \arg\max_k \left\{ \sum_{s \in S^{PO}} b(s) K_t^a(o|s) \sum_{u \in S^{PO}} P_t^{(a,o)}(u|s) \alpha_{t+1}^k(u) \right\} \tag{7.6}
$$

By combining Theorem 7.1 and Proposition 7.1 an explicit expression of the α-vectors can be derived. The algorithm that will be used utilizes this representation for solving the POMDP.

Proposition 7.2. *Let $\alpha_t^{l(b,a)}$ denote the maximizing α-vector for belief state b and action a at time t and let $\alpha_t^{l^*(b)}$ denote the optimizing α-vector for belief state b. Then the α-vectors can be expressed as:*

$$
\begin{aligned}
\alpha_t^{l(b,W)}(s) = & K_t^W(SD^-|s) \left[r_t(s,W,SD^-) + \sum_{u \in S^{PO}} P_t^{(W,SD^-)}(u|s) \alpha_{t+1}^{i(b,W,SD^-)}(u) \right] \\
& + K_t^W(SD^+|s) \left[r_t(s,W,SD^+) \right. \\
& \left. + \max_k \left\{ \sum_{u \in S^{PO}} P_t^{(W,SD^+)}(u|s) \alpha_{t+1}^k(u) \right\} \right] \qquad \text{if } s = 1, \\
& K_t^W(SD^-|s) \left[r_t(s,W,SD^-) + \sum_{u \in S^{PO}} P_t^{(W,SD^-)}(u|s) \alpha_{t+1}^{i(b,W,SD^-)}(u) \right]
\end{aligned}
$$

$$+ K_t^W(SD^+|s)R_t(s,W) \qquad \text{if } s = 2,3,4. \tag{7.7}$$

and

$$\begin{aligned}\alpha_t^{l(b,M)}(s) = {} & K_t^M(M^-|s)\left[r_t(s,M,M^-) + \sum_{u\in S^{PO}} P_t^{(M,M^-)}(u|s)\alpha_{t+1}^{i(b,M,M^-)}(u)\right] \\ & + K_t^M(M^+|s)\left[r_t(s,M,M^+)\right. \\ & \left. + \max_k\left\{\sum_{u\in S^{PO}} P_t^{(M,M^+)}(u|s)\alpha_{t+1}^k(u)\right\}\right] \qquad \text{if } s = 1, \\ & K_t^M(M^-|s)\left[r_t(s,M,M^-) + \sum_{u\in S^{PO}} P_t^{(M,M^-)}(u|s)\alpha_{t+1}^{i(b,M,M^-)}(u)\right] \\ & + K_t^M(M^+|s)R_t(s,M) \qquad \text{if } s = 2,3,4. \end{aligned} \tag{7.8}$$

Where

$$\begin{aligned} l^*(b) &= \arg\max_k\left\{\sum_{s\in S^{PO}} b(s)\alpha_t^k(s)\right\} \\ &= \arg\max_{\{l(b,W),l(b,M)\}}\left\{\sum_{s\in S^{PO}} b(s)\alpha_t^{l(b,W)}(s), \sum_{s\in S^{PO}} b(s)\alpha_t^{l(b,M)}(s)\right\} \end{aligned} \tag{7.9}$$

7.2.3 Algorithm

The algorithm we use is based on the fact that the optimal value function V^* is piecewise linear and convex. The idea was first described by Smallwood and Sondik [7] and later Monahan [13] and Lovejoy [14] simplified the algorithm. The basic outline of the algorithm is that first all possible α-vectors are generated using Eqs. (7.7) and (7.8), then non-optimal α-vectors are deleted and finally the optimal value function is constructed using the remaining α-vectors and the expression of $V_t^*(b)$ in Theorem 7.1, as can be seen in the pseudo code below.

Algorithm. Monahan's Algorithm with Eagle's Reduction

1. **Initialize.** $\alpha_T^1(s) = r_T(s)$, for all $s \in S^{PO}$, $A_T = \{\alpha^1\}$ and $t = T-1$
2. **Generate.** Generate $A_t = \{\alpha_t^1, \alpha_t^2 \cdots\}$ and mark all α-vectors.
3. **Eagle's reduction.**
   a. Select a marked α-vector α_t^i. If none exists go to step 4. Otherwise,
   b. Unmark the selected α-vector and if there exists an α_t^j such that $\alpha_t^i(s) \leq \alpha_t^i(s)$ for all $s \in S^{PO}$ delete the selected α-vector. Go to step 3(a)
4. **Monahan's elimination phase**
   a. Mark all the remaining α-vectors in A_t.
   b. Select a marked α-vector α_t^i. If none exists go to step 5. otherwise
   c. Unmark the α-vector and solve

$$\begin{aligned} \max \quad & \sigma \\ \text{s.t.} \quad & \sum_{s \in S^{PO}} b(s) \left(\alpha_t^i(s) - \alpha_t^j(s) \right) - \sigma \geq 0 \qquad \forall j \\ & \sum_{s \in S^{PO}} b(s) = 1 \\ & b(s) \geq 0 \qquad \forall s \in S^{PO}. \end{aligned} \tag{7.10}$$

   if the Linear Program (LP) yields a solution $\sigma \leq 0$, then remove the α-vector from A_t. Go to 3(b).
5. **Time update.** If $t > 1$, then $t = t-1$ and go to step 2, otherwise stop.

Generating the α-Vectors

Let $A_{t+1} = \{\alpha_{t+1}^1, \alpha_{t+1}^2, \cdots\}$ denote the set of α-vectors at time $t+1$. Now instead of determining the optimal α-vector $\alpha^{t^*(b)_t}$ by Eqs. (7.7)–(7.9) we generate the α-vector for every combination of an action and a vector α_{t+1}^i; denote this by $\alpha_t^{(a,i)}$. So we have

$$A_t = \left\{ \alpha_t^{(W,i)}, \alpha_t^{(M,i)} \right\}_{i=1\cdots||A_{t+1}||} \tag{7.11}$$

with

$$\begin{aligned}
\alpha_t^{(W,i)}(s) = & \sum_{o\in\Theta^W} K_t^W(o|s) \left[r_t(s,W,o) + \sum_{u\in S^{PO}} P_t^{(W,o)}(u|s)\alpha_{t+1}^i(u) \right], \\
& \text{if } s = 1, \\
& K_t^W(SD^-|s) \left[r_t(s,W,SD^-) + \sum_{u\in S^{PO}} P_t^{(W,SD^-)}(u|s)\alpha_{t+1}^i(u) \right] \\
& + K_t^W(SD^+|s) R_t(s,W) \qquad \text{if } s = 2,3,4. \\
\alpha_t^{(M,i)}(s) = & \sum_{o\in\Theta^M} K_t^M(o|s) \left[r_t(s,M,o) + \sum_{u\in S^{PO}} P_t^{(M,o)}(u|s)\alpha_{t+1}^i(u) \right], \\
& \text{if } s = 1, \\
& K_t^M(M^-|s) \left[r_t(s,M,M^-) + \sum_{u\in S^{PO}} P_t^{(M,M^-)}(u|s)\alpha_{t+1}^i(u) \right] \\
& + K_t^M(M^+|s) R_t(s,M) \qquad \text{if } s = 2,3,4.
\end{aligned} \tag{7.12}$$

Monahan's Elimination and Eagle's Reduction Phase

A vector α_t^i is called completely dominated if there exists another vector α_t^j such that $\alpha_t^i(s) \leq \alpha_t^j(s)$ for all $s \in S^{PO}$. Since the optimal value function at a certain belief state, $V_t^*(b)$, can be written as the maximum over the set of α-vectors A_t (Theorem 7.1) it follows immediately that if an α-vector is completely dominated it is never part of the optimal value function. The procedure where the α-vectors are checked for complete dominance is known as Eagle's reduction phase [15].

After this there may be α-vectors which are not completely dominated by the other vectors but still are dominated in the sense that they are never part of the optimal value function. Monahan [13] has shown that if, for a certain α-vector the optimal solution of the LP in (7.10) is non-positive we can remove it from the set A_t. Eagle [15] notes in his paper that this reduction of the number of α-vectors, which is known as the Monahan reduction phase, is not necessary when the number of actions, observations and decision epochs is not too large.

When all the α-vectors are generated for every decision epoch and the (completely) dominated ones are deleted the optimal value function follows directly from (7.4), and because every α-vector has an action associated with it (7.12), the optimal action is easy to determine.

Parameter	Source
Probability of death	CBS [16]
State transitions in S^{PO}	NCR [17]
Disutility of a mammogram	Mandelblatt et al. [18]
Disutility of a biopsy	Velanovich [19]
Specificity and sensitivity of mammography	Kolb et al. [20]
Specificity and sensitivity of self-detection	ibid.
Survival rates	NCR [17]
Life expectancy	CBS [16]
Value of life	WHO [21]

Table 7.1: Sources of model parameters

7.3 Model Parameters

In this section we present the input parameters and their sources. Table 7.1 provides a list with the sources of the model parameters. For each set of patient characteristics the parameters will differ. In the next section we present the results for several groups of patients based on their age and the grade of differentiation of the primary tumour. However for each set of group characteristics, as long as the parameters are available, the model can be applied.

The probability that a patient dies between two decision epochs depends only on the age of the patient and is obtained from Statistics Netherlands (Centraal Bureau voor de Statistiek) [16]. Whenever the age of patients in a certain group differs we will use the probability of death for the average age, e.g. when the age in a group is between 40 and 50 we use the probability of death of a 45 year old woman.

The state transition probabilities between the partially observable states, i.e. the probability that a patient gets a SP tumour or a LRR between two decision epochs, are obtained from the Netherlands Cancer Registry (NCR) [17]. The estimations of the disutility of a mammogram vary between 0.5 and 1.5 days [18], so we use an estimate of one day. The disutility of a biopsy is estimated between 2–4 weeks [19], in our model we take the average of three weeks. We assume that these disutilities are the same for all ages. The specificity and sensitivity of both mammography and self-detection are obtained from Kolb et al. [20].

The LSR and the end rewards are based on the life expectancy of a healthy patient. The expected remaining life years of an average patient at the start of the follow-up and at the end are used to construct a linear function of the life expectancy of a healthy patient at time $t = 1 \cdots T$. The expected remaining life years for patients in the different cancer states, i.e. the LSR and the end rewards, are calculated as a percentage of the expected remaining life years of a healthy patient. These percentages are based on the ten-year survival rates for the different groups, which are also obtained from the NCR [17].

For several of the input parameters the precise values are not available. The core state transitions, for instance, are based on the current policy of annual mammography. The probabilities will therefore be slightly shifted in time, e.g. if in reality a patient is most likely to get a LRR after 14 months it will not be detected for at least ten months when the next mammogram is taken, so the transition probabilities will suggest a later time at which the patient is most likely to get a LRR. Also it is very hard to give a precise estimation of the probability that an early detectable LRR will turn into a late detectable LRR between two decision epochs. In Sect. 7.5 we will discus to which extend these inaccuracies will affect the outcome. A complete list of the parameters used can be found in the appendix.

Average healthy life expectancy (years)	39.44
LSR early LRR	86 %
LSR late LRR	69 %
LSR SP	80 %

Table 7.2: LSRs as a percentage of the healthy life expectancy

7.4 Results

Since the optimal policy will vary for patients with different characteristics, in this section we present the results for one specific stratification of the patients. This stratification serves as an illustration and the reader should bear in mind that the model can be applied to much more specified stratifications and subsequently more personalized follow-up. We stratify based on two personal risk factors of the patients, the age of the patient and the grade of differentiation of the primary tumour. For the age we take the group of patients upto 50 years old because they have the highest risk of a LRR of the different age categories [5]. The grade of differentiation is chosen as the second risk factor because Witteveen et al. [5] found it to be one of the main risk factors for a LRR. With this stratification we thus consider four groups of patients, the first group are all patients with age upto 50, the second, third and fourth groups are patients with age upto 50 and grade of differentiation 1, 2 and 3, respectively.

As explained in Sect. 7.3, the parameters that depend on the group characteristics are the LSRs, the life expectancy and the core state transitions. In Table 7.2 the life expectancies and the LSRs (as a percentage of the life expectancy) are given. Since the probability of getting cancer per time interval is small (≈ 0.01) and the specificity of both mammography and self-detection is high (≈ 0.99), the majority (approximately 85 %) of patients will never have a positive mammogram or a self-detection. We present the optimal policy given that the patient never has a positive mammogram or a self-detection. The optimal policies for these patients, for each of

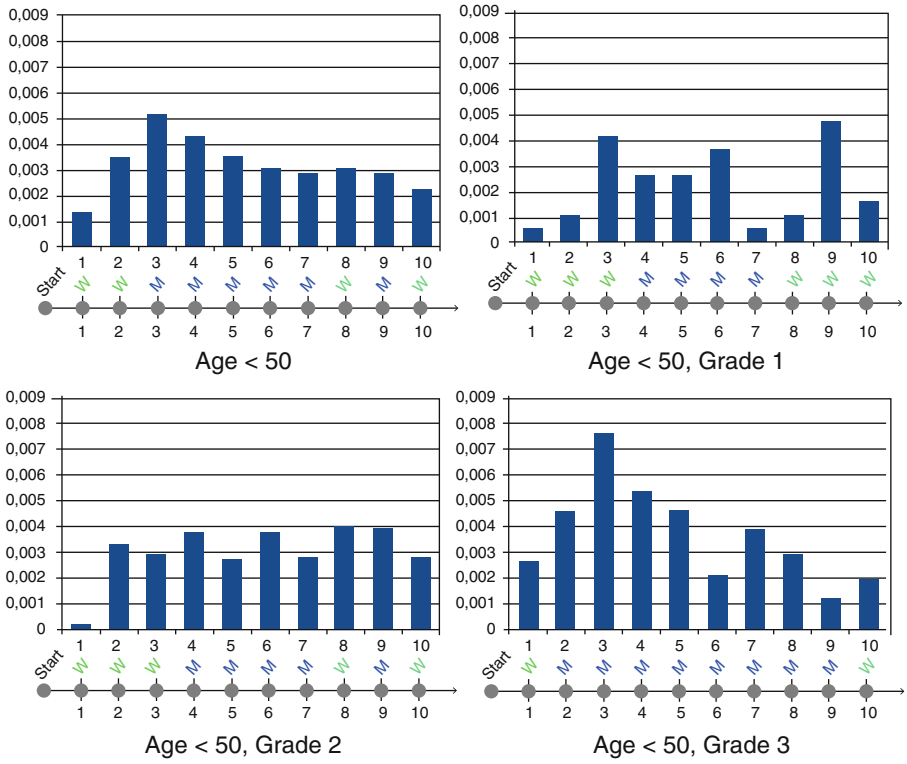


Fig. 7.2: Probability of a LRR and the optimal policy for each of the categories. Abbreviations: W = wait, M = mammography

the categories, are given in Fig. 7.2. (Of course, for any other series of observations the optimal policy can be obtained in a similar manner, but since there are 2^{10} different observation series possible we restrict ourselves to this case). Above the optimal policy the probability of a LRR is given. As one can see it is optimal to intensify the follow-up when the probability of a LRR peaks around the second year of follow-up and just after that.

The optimal policy for a patient will depend on the test results of mammograms previously taken and possible self-detections. For example: A patient, age 46 and differentiation grade 1, decides to wait 1.5 years ($t = 3$) after initial treatment ended, as recommended by the optimal policy (see Fig. 7.2). Now in case she does a false positive self-detection, the optimal policy will change and it will be optimal to skip the next mammogram (at $t = 3$).

If the initial belief about the patient's health is different, the optimal policy will be different too. For instance if the belief that the true health state of the patient is one of the cancer states is 0.03 instead of 0.0039 (which is the belief following from the actual data) it is optimal to test sooner and more often to neutralize the initial belief. In Fig. 7.3 the adjusted optimal policy is compared with the previous one, under the assumption that the patient does not have positive mammograms or self-detection. This example shows that it is beneficial to adjust the policy when the belief about the true health state of the patient changes.

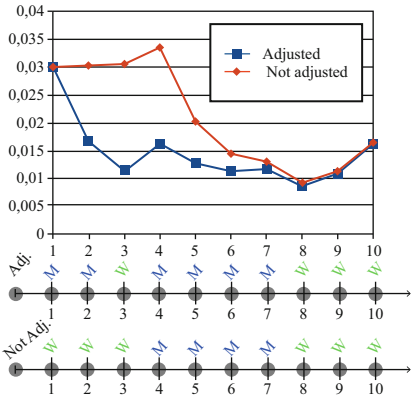


Fig. 7.3: Comparison in belief of cancer between the adjusted and unadjusted policy when the initial belief is high. Abbreviations: W = wait, M = mammography

It is not straightforward to determine the expected number of mammograms an average patient will have during the follow-up. It is however straightforward to provide an upper bound on this number. When a patient has a positive mammogram or a self-detection she will undergo a biopsy. If the biopsy is positive, the patient's decision process will stop. Otherwise, the belief state of the patient will be that the patient is with probability 1 healthy, because a biopsy is assumed to be a perfect test. With this belief state it is always optimal to select wait as the action at the next decision epoch. The number of mammograms stated in Fig. 7.2 is therefore an upper bound on the expected number of mammograms.

In Fig. 7.4 the total belief of cancer, that is the belief that the true health state of the patient is an early detectable LRR, a late detectable LRR or SP, $s \in \{2,3,4\}$, is given for the four groups when the policy is the optimal, the annual follow-up or the no follow-up policy. Again this is under the assumption that the observation in

case of Mammogram is negative and in case of Wait is no self-detection. Based on these figures we can conclude that the probability of cancer does not depend on the distribution of the mammograms but on the number of mammograms.

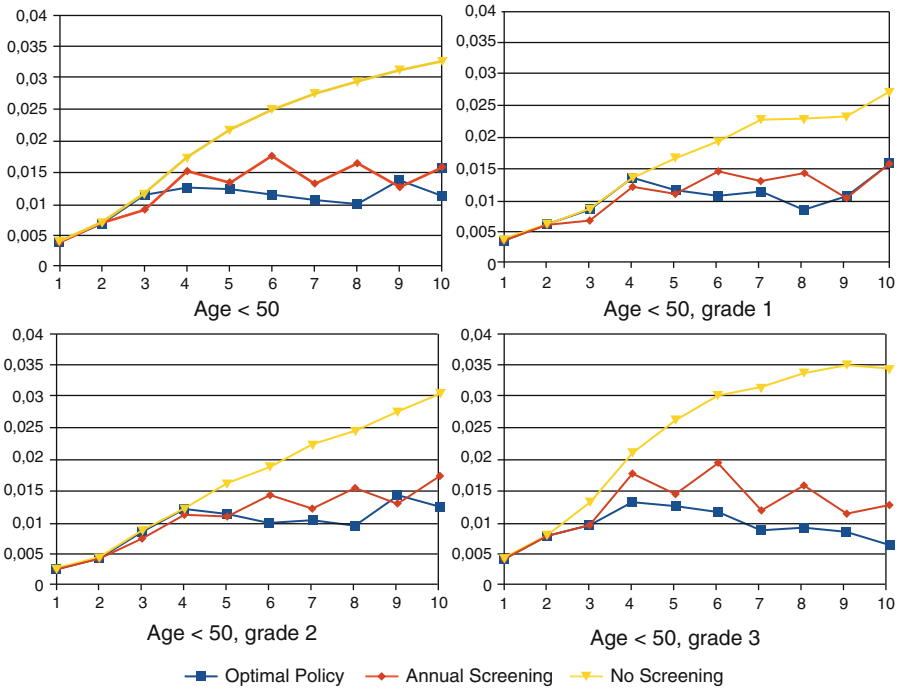


Fig. 7.4: Belief state at time $t = 1 \cdots T$ for different policies

In Table 7.3 the absolute and relative group sizes are given. Using these numbers and combining these with the upper bound on the number of mammograms, a lower bound on the difference in number of mammograms, when the annual policy is exchanged for the optimal policy can be established. The gain in expected QALYs is also given in Table 7.3. In the optimal policy a patient will have, on average, 1 mammogram more than in the current policy but the expected gain of QALYs amounts to 511.3. Also for the grade 3 patients a lot is to gain if the follow-up is intensified.

Age	Patients	Gain QALYs per patient	Total gain (QALYs)	Gain mmg per patient	Total Gain (mammograms)
<50	12,058 (100 %)	0.0424	511.3	−1	−12,058
<50, grade 1	1943 (16.11 %)	0.0229	44.5	1	1943
<50, grade 2	4847 (40.20 %)	0.0312	151.2	0	0
<50, grade 3	5268 (43.69 %)	0.0629	331.4	−3	−15,804

Table 7.3: Gain for the optimal policy in expected QALYs and number of mammograms for the different groups relative to the annual policy

A Societal Perspective

Instead of approaching the problem from the patient's perspective, we can also approach it from societal perspective. By ascribing a value to a life year (denoted as value of a statistical life year, VOSL) and replacing disutilities by the costs of a mammogram and a biopsy we can solve the problem by the same algorithm. The objective then is to maximize rewards instead of QALYs.

In our analysis of the adjusted problem we found that there were no differences in optimal policies between the optimization of the rewards and of the QALYs. This is due to the fact that for both a mammogram and a biopsy the ratio life year/disutility is roughly equal to the ratio value of a life year/costs.

7.4.1 Sensitivity Analyses

For some input parameters the only available estimations are rather rough. It is therefore insightful to analyse to which extent the optimal policy is sensitive to small changes in these parameters. From the results displayed in Fig. 7.2 it follows that the distribution of the mammograms depends heavily on the distribution of the probability of a LRR. It is therefore helpful to use the number of mammograms in the optimal policy as a measure to compare different policies instead of the distribution of the mammograms in the follow-up phase.

We analyse the sensitivity of the optimal number of mammograms to two parameters. Firstly we analyse the sensitivity to the LSRs. We find that the optimal policy is not very sensitive to how much percent of the healthy life expectancy the LSR is, but rather to the difference in LSRs between an early detectable LRR and a late detected LRR. We increase the difference by 2, 4 and 6 percent point and decrease it with the same amount and denote the number of mammograms in the optimal policy. The results are presented in Table 7.4. It turns out that the optimal number of mammograms in any group already changes when the difference in LSR is altered by only 2 percent point.

Age, grade		<50, −	<50, 1	<50, 2	<50, 3
Difference in LSR between early and late LRR relative to Table 7.2 (percent point)	+6	7	5	8	8
	+4	7	5	6	8
	+2	7	5	6	8
	0	6	4	5	8
	−2	6	4	5	7
	−4	5	3	5	6
	−6	4	3	4	5
Growth rate LRR	×0.5	4	3	3	5
	×1	6	4	5	8
	×1.5	7	5	8	8
	×2	8	6	8	9
	×3	9	6	8	9

Table 7.4: Sensitivity of the optimal number of mammograms for different parameters

Secondly we analyse the sensitivity to the rate at which an early detectable LRR grows into a late detectable LRR. In order to do this we determine the number of mammograms in the optimal policy when the LRR growth rate is 0.5 times, 1.5 times, 2 times and 3 times the normal value. The results are also presented in Table 7.4. Here also it appears that the optimal result is quite sensitive to changes in the growth rate of a LRR.

The main conclusion we can draw from these results is that the optimal result is quite sensitive to changes in the LSR and the growth rate of a LRR. We can also conclude that this effect is somewhat stronger for patients with grade 1 than for patients with grade 2 or 3.

7.5 Conclusions and Discussion

Currently breast cancer patients in the Netherlands have annual check-ups after treatment. Even though it is known that many factors, such as age, characteristics of the primary tumour and treatment to detect LRRs and SP tumors in an early phase are of great influence on the risk of a LRR, the follow-up is the same for all patients. Individual mammography follow-up decisions based on personal risk characteristics are proposed by the national guideline of the Netherlands but without results in practice. In this chapter we develop an analytic model to personalize the decision making based on personal risk characteristics.

We formulate this problem as a partially observable Markov decision process. Our results show that, indeed, the optimal follow-up policy depends heavily on the personal risk characteristics of a patient. For all patients it is optimal to intensify the follow-up when the probability of a LRR peaks and just after instead of uniformly distributing the check-ups over the follow-up phase. Our results furthermore show

that the personal test history of a patient is very important for determining the optimal test policy. Another conclusion that we can draw from our results is that instead of testing all patients annually it is more efficient to test patients with differentiation grade 3 more often and with differentiation grade 1 less often, for patients with differentiation grade 2 our model suggest to leave the number of mammograms the same. Finally, the results show that moving from a patient's perspective to a societal perspective does not change the optimal policy. This is an important result because in the case of actual implementation of the results in practice one does not need to choose between the benefit of an individual and that of society as a whole.

Implementation of the optimal policy has a few advantages for both the patient and society. From the patient's perspective an advantage is that the expected number of QALYs increases. Another advantage is that the number of mammograms, although on average it increases, decreases for the group of patients for which it seems unnecessary to have a mammogram that often. This is a big advantage because having a mammogram is a rather unpleasant experience [22]. From society's perspective the main advantage is that a lot of QALYs can be gained when allocating the resources to where it is needed most.

The main advantage of this model compared to current follow-up policies is that when good estimates of the model parameters are available, an optimal follow-up policy can be determined based on personal risk characteristics which can be updated at each decision epoch based on the observation made. When the parameters are available it is fairly straightforward to implement the algorithm into a tool which can than be used by a physician to tailor the follow-up to the patient's personal characteristics.

The biggest limitation of our study is that the estimates for some of the model parameters are quite inexact, in particular the survival rates used to estimate the LSR and the rate at which an early detectable LRR develops into an late detectable LRR. Sensitivity analyses for these parameters show that the optimal policy is quite sensitive to these parameters. Therefore, without further study to obtain better estimates for these model parameters, the model cannot be used to give recommendations about the exact testing policies.

Another limitation of our model is that it is a discrete model. This has two disadvantages compared to a continuous model. Firstly, in a continuous model the actual time that a patient has a LRR could be taken into account instead of the rather arbitrary distinction between an early detectable LRR and a late detectable LRR. The LSR could then be much more precise. Secondly, in a continuous model a mammogram can be taken at an arbitrary time. Our recommendation for further study is therefore to develop a decision model in which these two variables can be modelled continuously instead of discretely.

Appendix: Notation

General	This chapter	Description
S	S	State space
s	s_t	State at time t
r	$r_t(s,a,o)$	Reward at time t in state s with action a and observation o
A	A'_t	Set of actions at time t
a	a_t	Action at time t
P	$P_t^{(a,o)}$	Transition probability at time t given action a and observation o
$V_t^*(s)$	$V_t^*(\pi)$, $V_t^*(b)$	Optimal value function from time t onwards given information state π or belief state b

References

1. W.L. Lu, L. Jansen, W.J. Post, J. Bonnema, J.C. van de Velde, G.H. De Bock, Impact on survival of early detection of isolated breast recurrences after the primary treatment for breast cancer: a meta-analysis. Breast Cancer Res. Treat. **114**, 403–412 (2009). http://dx.doi.org/10.1007/s10549-008-0023-4
2. M. Moossdorff, L.M. van Roozendaal, L.J.A. Strobbe, S. Aebi, D.A. Cameron, J.M. Dixon, A.E. Giuliano, B.G. Haffty, B.E. Hickey, C.A. Hudis, V.S. Klimberg, B. Koczwara, T. Kühn, M.E. Lippman, A. Lucci, M. Piccart, B.D. Smith, V.C.G. Tjan-Heijnen, C.J.H. van de Velde, K.J.V. Zee, J.B. Vermorken, G. Viale, A.C. Voogd, I.L. Wapnir, J.R. White, M.L. Smidt, Maastricht Delphi consensus on event definitions for classification of recurrence in breast cancer research. J. Natl. Cancer Inst. **106**(12), 1–7 (2014). http://dx.doi.org//10.1093/jnci/dju288
3. IKNL, Dutch Breast Cancer Guideline (2016), available: https://www.oncoline.nl/ [Online]. Accessed 2 March 2016
4. S.M.E. Geurts, F. de Vegt, S. Siesling, K. Flobbe, K.K.H. Aben, M. van der Heiden-van der Loo, A.L.M. Verbeek, J.A.A.M. van Dijck, V.C.G. Tjan-Heijnen, Pattern of follow-up care and early relapse detection in breast cancer patients. Breast Cancer Res. Treat. **136**, 859–868 (2012). http://dx.doi.org/10.1007/s10549-012-2297-9
5. A. Witteveen, I.M.H. Vliegen, G.S. Sonke, J.M. Klaase, M.J. IJzerman, S. Siesling, Personalisation of breast cancer follow-up: a time-dependent prognostic nomogram for the estimation of annual risk of locoregional recurrence in early breast cancer patients. Breast Cancer Res. Treat. **152**, 627–636 (2015). http://dx.doi.org/10.1007/s10549-015-3490-4
6. M. IJzerman, A. Manca, J. Keizer, S. Ramsey, Implementing comparative effectiveness research in personalized medicine applications in oncology: current and future perspectives. Comp. Eff. Res. **26**(5), 65–72 (2015). https://dx.doi.org/10.2147/CER.S92212

7. R.D. Smallwood, E.J. Sondik, The optimal control of partially observable Markov processes over a finite horizon. Oper. Res. **21**(5), 1071–1088 (1973). http://dx.doi.org/10.1287/opre.21.5.1071
8. L.N. Steimle, B.T. Denton, Markov decision processes for screening and treatment of chronic diseases, in *Markov Decision Processes in Practice*, ed. by R. Boucherie, N.M. van Dijk (Springer, New York, 2016)
9. T. Ayer, O. Alagoz, N.K. Stout, A POMDP approach to personalize mammography screening decisions. Oper. Res. **60**(5), 1019–1034 (2012). http://dx.doi.org/10.1287/opre.1110.1019
10. M.U.S. Ayvaci, O. Alagoz, E.S. Burnside, The effect of budgetary restrictions on breast cancer diagnostic decisions. MSOM **14**(4), 600–617 (2012). http://dx.doi.org/10.1287/msom.1110.0371
11. J. Zhang, B.T. Denton, H. Balasubramanian, N.D. Shah, B.A. Inman, Optimization of PSA screening policies: a comparison of the patient and societal perspectives. Med. Decis. Making **32**(1), 337–349 (2012). http://dx.doi.org/10.1177/0272989X11416513
12. F.A. Sonnenberg, J.R. Back, Markov models in medical decision making, a practical guide. Med. Decis. Making **13**(4), 322–338 (1993). http://dx.doi.org/10.1177/0272989X9301300409
13. G.E. Monahan, A survey of partially observable Markov decision processes: theory, models and algorithms. Manag. Sci. **28**(1), 1–16 (1982). http://dx.doi.org/10.1287/mnsc.28.1.1
14. W.S. Lovejoy, A survey of algorithmic methods for partially observed Markov decision processes. Ann. Oper. Res. **28**(1), 47–65 (1991). http://dx.doi.org/10.1007/BF02055574
15. J.N. Eagle, The optimal search for a moving target when the search path is constrained. Oper. Res. **32**(5), 1107–1115 (1984). http://www.jstor.org/stable/170656
16. CBS, Statline (2016), available: http://statline.cbs.nl/Statweb/ [Online]. Accessed 18 May 2016
17. Netherlands Comprehensive Cancer Organisation (IKNL), Netherlands Cancer Registry (2016), available: https://www.cijfersoverkanker.nl/ [Online]
18. J.S. Mandelblatt, M.E. Wheat, M. Monane, R.D. Moshief, J.P. Hollenberg, J. Tang, Breast cancer screening for elderly women with and without comorbid conditions: a decision analysis model. Ann. Internal Med. **116**(9), 722–730 (2002). http://dx.doi.org/10.7326/0003-4819-116-9-722
19. V. Velanovich, Immediate biopsy versus observation for abnormal findings on mammograms: an analysis of potential outcomes and costs. Am. J. Surg. **170**(4), 327–332 (1995). http://dx.doi.org/10.1016/S0002-9610(99)80298-0
20. T.M. Kolb, J. Lichy, J.H. Newhouse, Comparison of the performance of screening mammography, physical examination, and breast us and evaluation of factors that influence them: an analysis of 27,825 patient evaluations. Radiology **225**, 165–175 (2002). http://dx.doi.org/10.1148/radiol.2251011667
21. WHO, The world health report: 2002: reducing risks, promoting healthy life. World Health Organization (2002)
22. M. Fine, B. Rimer, P. Watts, Women's responses to the mammography experience. J. Am. Board Fam. Pract. **6**(6), 546–555 (1993)

Chapter 8
Advance Patient Appointment Scheduling

Antoine Sauré and Martin L. Puterman

Abstract This chapter describes the use of the linear programming approach to approximate dynamic programming as a means of solving advance patient appointment scheduling problems, which are problems typically intractable using standard solution techniques. Starting from the linear programming approach to discounted infinite-horizon Markov decision processes, and employing an affine value function approximation in the state variables, the method described in this chapter provides a systematic way of identifying effective booking guidelines for advance patient appointment scheduling problems. Two applications found in the literature allow us to show how these guidelines could be used in practice to significantly increase service levels for medical appointments, measured as the percentage of patients booked within medically acceptable wait times, and thus to decrease the potential impact of delays on patients' health.

8.1 Introduction

Globally, health care systems are facing increasing and lengthy wait times for a wide range of services. This is partially attributable to the effects of an increasing demand for care, constrained public and private budgets, and diminishing supply of staff. Although in some cases unnecessary delays have little medical impact on patients,

A. Sauré (✉)
Telfer School of Management, University of Ottawa, 55 Laurier Ave E, Ottawa, ON, Canada K1N 6N5
e-mail: asaure@uottawa.ca

M.L. Puterman
Sauder School of Business, University of British Columbia, 2053 Main Mall, Vancouver, BC, Canada V6T 1Z2
e-mail: martin.puterman@sauder.ubc.ca

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_8

in others, they may deteriorate patients' health. For this reason, health care providers and policy makers around the world are under constant political and community pressure to improve efficiency and reduce wait times for medical services.

In this context, the development and implementation of effective patient appointment scheduling methodologies has become critical for health care organizations to improve patient flow and provide timely access to care. Through an improved match between the availability of limited medical resources and patient needs, and a better assessment of the future demand for services, sophisticated patient appointment scheduling methodologies have proved to be an effective way of reducing patient wait times for medical appointments and increasing the utilization of critical resources, influencing the overall performance of health care systems.

Patient appointment scheduling methodologies are usually classified as either allocation scheduling or advance scheduling [19]. Allocation scheduling refers to methodologies for assigning specific appointment times to patients, but only once all patients for a given service day have been identified. Allocation scheduling methods typically assume that a first level of scheduling is required which assigns patients to specific service days and resources. Advance scheduling, on the other hand, refers to methodologies for scheduling patient appointments in advance of the service day, when the future demand for a particular service is still unknown. Advance scheduling methods usually assume that a second level of scheduling is needed which assigns patients to specific appointment times. Although the implied problem decomposition is not necessarily optimal, no dynamic patient appointment scheduling methodologies have been developed to address the two levels of scheduling simultaneously. Most studies in the patient appointment scheduling literature address allocation scheduling problems. Magerlein and Martin [19], Cayirli and Veral [8], Mondschein and Weintraub [20], Gupta and Denton [16], Cardoen et al. [7] and Begen and Queyranne [3] provide comprehensive reviews on this topic. The methodology described in this chapter deals with a general advance patient appointment scheduling problem. This class of problems has received limited attention in the literature. Patrick et al. [21], Erdelyi and Topaloglu [14], Schütz and Kolisch [26], Sauré et al. [24], Gocgun and Puterman [15] and Sauré et al. [25] are some of the few papers published on this topic.

Advance patient appointment scheduling decisions typically rely on the expertise of one or two bookings agents who often make them without explicitly considering their impact on the future performance of the system. The presence of a highly variable number of appointment requests, limited service capacity and multiple types of patients makes it extremely challenging for the agents to adequately assess the real impact of their actions in order to more efficiently allocate capacity. This unintended lack of foresight generates several inefficiencies that usually translate into unnecessary delays, an unsystematic prioritization of patients and inefficient resource utilization.

By adequately assessing the future impact of advance patient appointment scheduling decisions, the booking policies derived using the methodology described in this chapter, although possibly sub-optimal, perform significantly better than myopic policies. The latter are policies that only consider the immediate cost of the appointment scheduling decisions, ignoring the potential cost-to-go values associated with them. In other words, they only consider what is known at the

present, ignoring the future impact of today's decisions. Myopic policies are usually representative of current practices.

The booking policies derived using the methodology described in this chapter provide clear and effective guidelines as to when to book patients and when to resort to an alternative source of capacity such as overtime. We will see, through two applications found in the literature, how these guidelines could be used in practice to significantly increase service levels for medical appointments, measured as the percentage of patients booked within medically acceptable wait times, and thus to decrease the potential impact of delays on patients' health.

8.2 Problem Description

Consider a single medical resource with a daily regular-hour capacity that has been divided into C_r fixed-length appointment slots. Each day the booking agent in charge of the medical resource receives appointment requests from I pre-specified patient types. Patients are classified into types according to their priorities and capacity requirements, each priority i having a different medically acceptable wait time (in days), also called wait time target, denoted by T_i. The capacity requirements associated with a patient of type i are represented by a vector $\mathbf{r}_i = \{r_{ij}\}_{j=1}^{l_i}$, where l_i denotes the number of days over which the patient's appointments are scheduled and r_{ij} denotes the duration, in number of appointment slots, of the patient's appointment on day j of this time period. For example, $r = 1$ represents one session of one appointment slot, $\mathbf{r} = (1,0,0,0,0,1)$ represents an initial session of one appointment slot and a follow-up session also of one appointment slot five working days after, and $\mathbf{r} = (2,1,1,1,1)$ represents a medical treatment consisting of an initial session of two appointment slots followed by four daily consecutive sessions of one appointment slot each. We assume that the service times are deterministic, and measured in number of appointment slots, and that the number of appointment requests of type i the booking agent receives on any given day follows an independent discrete probability distribution with mean m_i. Patient types and demand distributions do not change over time and demand is assumed uncorrelated over patient types.

Advance patient appointment scheduling decisions are made once a day and the booking agent can schedule patients' first appointments at most N days in advance. To relieve excess demand, the booking agent has the ability to use overtime at a cost of h per appointment slot. The overtime capacity is C_o appointment slots per day. The penalty associated with booking the first appointment of a patient of type i on day n (from today) is denoted by c_{in}. We assume that c_{in} is non-decreasing in n and equal to zero if $n \leq T_i$. The values of c_{in}, although arbitrary, are used in our setting to model different wait time targets. However, they can also be used to model other types of piecewise linear functions such as, for example, time windows (i.e., wait time targets plus and minus a few days) (see [15]). As an alternative action,

the booking agent may also decide to postpone the scheduling decisions for some patients. The penalty associated with delaying the scheduling decisions for a patient of type i is denoted by g_i. Neither rescheduling nor cancellations are considered.

Figure 8.1 depicts the timeline associated with the booking agent's task. At the beginning of each day, the booking agent observes the number of appointment slots already booked on each of the subsequent M days. Since the agent can schedule patients' first appointments at most N days in advance, $M \geq N + \max_i\{l_i\} - 1$ to allow the booking of all the appointments for patients scheduled starting on day N. Then the booking agent waits until the end of the day to learn the number of patients of each type waiting to be booked and decides how to allocate the available appointment slots among the different patient types.

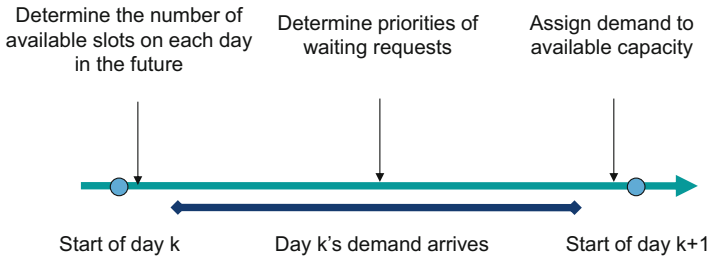


Fig. 8.1: Timeline associated with the advance patient appointment scheduling decisions

8.3 Mathematical Formulation

The advance patient appointment scheduling problem described above can be formulated as the following discounted infinite-horizon Markov Decision Process (MDP). The main objective of this model is to identify effective ways of allocating available regular-hour and overtime capacity to incoming appointment requests while increasing service levels in a cost-effective manner.

8.3.1 Decision Epochs

Decisions are made at the end of each day over an infinite time horizon.

8.3.2 State Space

At the end of each day, the booking agent knows the number of regular-hour and overtime appointment slots already booked on each of the subsequent M days as well as the number of patients of each type waiting to be booked. Thus, a state of the system, denoted by $\mathbf{s} \in S$, is represented by a vector $\mathbf{s} = (\mathbf{u}, \mathbf{v}, \mathbf{w}) = (u_1, \ldots, u_M, v_1, \ldots, v_M, w_1, \ldots, w_I)$, where u_m is the number of regular-hour appointment slots already booked on day m (from today), v_m is the number of overtime appointment slots already booked on day m (from today) and w_i is the number of patients of type i waiting to be booked. As a consequence of considering an M-day rolling booking horizon, the number of appointment slots booked on day M of any valid state of the system is equal to zero. That is $u_M = v_M = 0$. Figure 8.2 illustrates the relationship between the decision epochs and the finite booking horizon. All states are non-negative and integer.

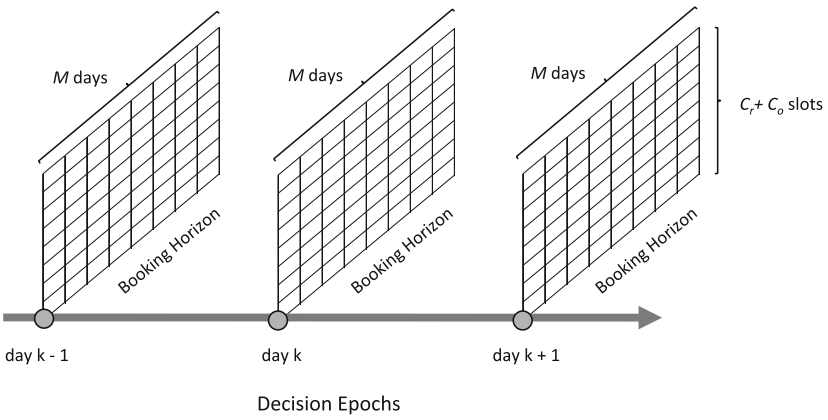


Fig. 8.2: Relationship between the decision epochs and the finite rolling booking horizon. At the beginning of each day, the booking agent observes the number of appointment slots already booked on each of the subsequent M days. Day m of the booking horizon at the current decision epoch becomes day $m-1$ of the booking horizon at the subsequent decision epoch

8.3.3 Action Sets

At the end of each day, the booking agent must decide on which day to book the first appointment for each of the patients waiting to be booked. In some cases, this implies the use of overtime. Alternatively, the booking agent may postpone to the next day the scheduling decisions for some patients. Thus, an action

available to the booking agent, denoted by $\mathbf{a}$, is represented by a vector $\mathbf{a} = (\mathbf{x}, \mathbf{y}) = (x_{11}, \ldots, x_{IN}, y_1, \ldots, y_M)$, where x_{in} is the number of patients of type i whose first appointment is booked on day n (from today) and y_m is the number of overtime appointments slots required on day m (from today). Note that rather than assigning patients to specific appointment slots, the model allocates available daily capacity to each patient type. Once daily capacity is allocated, a second level of scheduling is needed to assign patients to specific appointment slots.

The set of feasible actions compatible with state $(\mathbf{u}, \mathbf{v}, \mathbf{w}) \in S$, denoted by $A_{(\mathbf{u},\mathbf{v},\mathbf{w})}$, must satisfy the following constraints:

$$\sum_{n=1}^{N} x_{in} \leq w_i \qquad \forall i \qquad (8.1)$$

$$u_m + \sum_{i=1}^{I} \sum_{k=\max\{m-l_i+1,1\}}^{\min\{m,N\}} r_{i(m-k+1)} x_{ik} \leq C_r + y_m \qquad \forall m \qquad (8.2)$$

$$\sum_{i=1}^{I} \sum_{k=\max\{m-l_i+1,1\}}^{\min\{m,N\}} r_{i(m-k+1)} x_{ik} \geq y_m \qquad \forall m \qquad (8.3)$$

$$v_m + y_m \leq C_o \qquad \forall m \qquad (8.4)$$

Constraint (8.1) restricts the number of booking decisions associated with patients of type i to be less than or equal to the number of patients of type i waiting to be booked. Constraint (8.2) limits the total number of appointment slots booked today on day m (from today) to be less than or equal to the total available capacity that day. Constraint (8.3) restricts the number of overtime appointment slots booked today on day m (from today) to be less than or equal to the total number of appointment slots booked today on that day. Constraint (8.4) ensures that the number of overtime appointment slots booked today on day m (from today) is less than or equal to the available overtime capacity that day. All actions are non-negative and integer.

8.3.4 Transition Probabilities

Once actions are taken, the only source of uncertainty in the transition to the next state of the system is the number of new appointment requests of each type. As a result of choosing action $\mathbf{a} = (\mathbf{x}, \mathbf{y})$ in state $\mathbf{s} = (\mathbf{u}, \mathbf{v}, \mathbf{w})$, $\mathbf{a} \in A_{\mathbf{s}}$ and $\mathbf{s} \in S$, and having q_i new appointment requests from patients of type i, the state of the system the next day, denoted by $\mathbf{s}' = (\mathbf{u}', \mathbf{v}', \mathbf{w}')$, is given by the following probability distribution:

$$p(\mathbf{s}'|\mathbf{s},\mathbf{a}) = \begin{cases} \prod_{i=1}^{I} \Pr(q_i), & \text{if } \mathbf{s}' = (\mathbf{u}',\mathbf{v}',\mathbf{w}') \text{ satisfies equations (8.6) to (8.8);} \\ 0, & \text{otherwise.} \end{cases} \tag{8.5}$$

$$u'_m = \begin{cases} u_{m+1} - y_{m+1} + \sum_{i=1}^{I} \sum_{k=\max\{(m+1)-l_i+1,1\}}^{\min\{(m+1),N\}} r_{i[(m+1)-k+1]} x_{ik}, & m < M; \\ 0, & m = M. \end{cases} \tag{8.6}$$

$$v'_m = \begin{cases} v_{m+1} + y_{m+1}, & m < M; \\ 0, & m = M. \end{cases} \tag{8.7}$$

$$w'_i = w_i - \sum_{n=1}^{N} x_{in} + q_i \qquad \forall i \tag{8.8}$$

Equations (8.6) and (8.7) define the new number of regular-hour and overtime appointment slots booked on day m (from today) as a function of the number of appointment slots previously booked on day $m+1$ (from today) plus all the new bookings that affected that day. Equation (8.8) specifies the new number of patients of type i waiting to be booked as the number of appointment requests of type i not booked previously, that is $w_i - \sum_{n=1}^{N} x_{in}$, plus the new appointment requests from patients of type i. The term $\Pr(q_i)$ in Equation (8.5) represents the probability of having q_i new appointment requests from patients of type i.

8.3.5 *Immediate Cost*

The immediate cost $c(\mathbf{s},\mathbf{a})$ associated with choosing action $\mathbf{a} = (\mathbf{x},\mathbf{y})$ in state $\mathbf{s} = (\mathbf{u},\mathbf{v},\mathbf{w})$, $\mathbf{a} \in A_{\mathbf{s}}$ and $\mathbf{s} \in S$, comes from three sources: the resulting wait time penalties, the use of overtime, and the penalties associated with postponing to the next day the scheduling decisions for some patients.

$$c(\mathbf{s},\mathbf{a}) = \underbrace{\sum_{i=1}^{I}\sum_{n=1}^{N} c_{in}x_{in}}_{\text{wait time penalties}} + \underbrace{\sum_{m=1}^{M} h_m y_m}_{\text{overtime cost}} + \underbrace{\sum_{i=1}^{I} g_i \left(w_i - \sum_{n=1}^{N} x_{in} \right)}_{\text{postponement penalties}} \tag{8.9}$$

In Eq. (8.9), h_m is the discounted cost associated with using an overtime slot on day m (from today). That is $h_m = \alpha^{m-1} h \; \forall m$, where α is the discount factor.

8.3.6 Optimality Equations

The value function in this formulation, denoted by $v : S \to \mathbb{R}_0^+$, corresponds to the expected α-discounted cost over the infinite horizon. Given a stationary appointment scheduling policy π, which uses decision rule δ every decision epoch t, $v(\mathbf{s})$, the expected α-discounted cost associated with state $\mathbf{s} \in S$, can be determined as follows:

$$v(\mathbf{s}) \equiv \mathbb{E}_{\mathbf{s}}^{\pi}\left[\sum_{t=1}^{\infty} \alpha^{t-1} c(\mathbf{s}_t, \delta(\mathbf{s}_t))\right] \qquad \forall \mathbf{s} \in S \tag{8.10}$$

Clearly, we are not so much interested in determining the value function for a specific appointment scheduling policy as in finding an optimal stationary policy. To identify such a policy we consider the following form of Bellman's optimality equations:

$$v(\mathbf{s}) = \min_{\mathbf{a} \in A_{\mathbf{s}}} \left\{ c(\mathbf{s}, \mathbf{a}) + \alpha \sum_{\mathbf{s}' \in S} p(\mathbf{s}' | \mathbf{s}, \mathbf{a}) v(\mathbf{s}') \right\} \quad \forall \mathbf{s} \in S \tag{8.11}$$

The challenge is that even for very small problem settings the size of the state space and the size of the corresponding action sets make a direct solution to Eq. (8.11) impossible. The state variable $\mathbf{s} = (\mathbf{u}, \mathbf{v}, \mathbf{w})$ and the action variable $\mathbf{a} = (\mathbf{x}, \mathbf{y})$ have $(2M + I)$ and $(I \times N + M)$ dimensions, respectively. Assuming that w_i can take up to Q possible values, this means that we might have up to $(C_r + 1)^{M-1} \times (C_o + 1)^{M-1} \times Q^I$ different states and $Q^{I \times N} \times (C_o + 1)^{M-1}$ different, although not necessarily feasible, actions. Thus, the size of the state space and the size of the corresponding action sets increase exponentially with the number of patient types (I) and the number of days in the booking horizon (M).

8.4 Solution Approach

Fortunately, a number of sophisticated methods for dealing with the problem caused by the exponential increase of the state space and the number of available actions, called Approximate Dynamic Programming (ADP), have been developed in the last couple of decades [6, 22, 28]. The literature in this field focuses primarily on two types of methods: simulation-based algorithms and linear programming algorithms.

Linear programming algorithms for discounted infinite-horizon MDPs are based on formulating Bellman's optimality equations as an equivalent linear program [11] and on choosing an appropriate approximation architecture to represent the variables (value function) in it. This approach was initially introduced by Schweitzer and Seidmann [27] and has been recently reconsidered by de Farias and Van Roy [9, 10], Adelman and Mersereau [2], Patrick et al. [21], Desai et al. [13], Sauré et al. [24], Adelman and Klabjan [1] and Gocgun and Puterman [15]. An appropriately chosen

approximation architecture reduces the number of variables in the equivalent linear program. This is one of the reasons why most approximation architectures found in the literature are defined as a linear combination of a low-dimensional set of basis functions.[1]

Given a pre-selected set of basis functions, the linear programming approach to high dimensional discounted infinite-horizon MDPs can be summarized in the following five steps:

Step 1: Write the optimality equations in their linear programming form.

$$\max_{\mathbf{v}} \left\{ \sum_{\mathbf{s}\in S} \gamma(\mathbf{s}) v(\mathbf{s}) : c(\mathbf{s},\mathbf{a}) + \alpha \sum_{\mathbf{s}'\in S} p(\mathbf{s}'|\mathbf{s},\mathbf{a}) v(\mathbf{s}') \geq v(\mathbf{s}) \ \forall \mathbf{a} \in A_{\mathbf{s}}, \mathbf{s} \in S \right\} \tag{8.12}$$

In this model, $\gamma(\mathbf{s})$ represents the weight of state $\mathbf{s} \in S$ in the objective function. The solution to the equivalent linear program (8.12) is the same as the solution to the optimality equations when γ is strictly positive [12, 17].

Step 2: Write the value function as a linear combination of basis functions.

$$v(\mathbf{s}) = V_0 + \sum_{k=1}^{K} V_k \phi_k(\mathbf{s}) \qquad \forall \mathbf{s} \in S \qquad V_0, \ldots, V_K \in \mathbb{R} \tag{8.13}$$

In Eq. (8.13), each $\phi_k : S \to \mathbb{R}$ represents a basis function and the parameters $\{V_k\}_{k=1}^{K}$ are the basis function weights.

Step 3: Formulate the approximate equivalent linear program.

$$\max_{V_0,\ldots,V_K} \left\{ V_0 + \sum_{k=1}^{K} \mathbb{E}_{\gamma}[\phi_k] V_k \right\} \tag{8.14}$$

subject to

$$(1-\alpha)V_0 + \sum_{k=1}^{K} v_k(\mathbf{s},\mathbf{a}) V_k \leq c(\mathbf{s},\mathbf{a}) \qquad \forall \mathbf{a} \in A_{\mathbf{s}}, \mathbf{s} \in S$$

[1] A basis function is a mapping S to $\mathbb{R}$.

where

$$\mathbb{E}_{\gamma}[\phi_k] = \sum_{\mathbf{s}\in S} \gamma(\mathbf{s})\phi_k(\mathbf{s}) \qquad \forall k$$

$$v_k(\mathbf{s},\mathbf{a}) = \phi_k(\mathbf{s}) - \alpha \sum_{\mathbf{s}'\in S} p(\mathbf{s}'|\mathbf{s},\mathbf{a})\phi_k(\mathbf{s}') \qquad \forall k, \mathbf{a}\in A_{\mathbf{s}}, \mathbf{s}\in S$$

The approximate equivalent linear program (8.14) results from substituting Eq. (8.13) into (8.12) and from restricting γ to be an exogenous probability distribution over the initial state of the system. That is $\sum_{\mathbf{s}\in S}\gamma(\mathbf{s}) = 1$. Thus, the task becomes to determine the best values for the approximation parameters.

Step 4: Solve (8.14) via constraint sampling or column generation.

The approximate equivalent linear program (8.14) has a tractable number of variables, $(K+1)$, but still an intractable number of constraints. For this reason, it is often necessary to use techniques such as constraint sampling [10] or column generation [5, 18]. Unfortunately, column generation is limited to affine value function approximation architectures in the state variables. The solution to (8.14) determines $\{V_k^*\}_{k=0}^K$, the optimal values of the approximation parameters.

Step 5: Compute approximate optimal actions.

$$\delta^*(\mathbf{s}) \in \arg\min_{\mathbf{a}\in A_{\mathbf{s}}} \left\{ c(\mathbf{s},\mathbf{a}) + \alpha \sum_{k=1}^{K} V_k^* \left[\sum_{\mathbf{s}'\in S} p(\mathbf{s}'|\mathbf{s},\mathbf{a})\phi_k(\mathbf{s}') \right] \right\} \quad \forall \mathbf{s}\in S \quad (8.15)$$

In practice, rather than computing and storing the approximate optimal actions for each state $\mathbf{s}\in S$, a resource-intensive task, we only compute them as needed by solving (8.15). The minimization problem in (8.15) is obtained by inserting the approximate value function defined by $\{V_k^*\}_{k=0}^K$ into the right hand side of the optimality equations (8.11) and ignoring constant terms.

Although linear programming algorithms for this type of models are mostly limited to approximation architectures that are defined as a linear combination of a low-dimensional set of basis functions, their main advantage with respect to simulation-based algorithms is that they avoid the need for iterative learning and often provide good results.

To solve the advance patient appointment scheduling model described above, we chose (8.13), the approximation to $v(\mathbf{s})$ described in Step 2 of the solution approach, to be the following affine approximation:

$$v(\mathbf{u},\mathbf{v},\mathbf{w}) = W_0 + \sum_{m=1}^{M} U_m u_m + \sum_{m=1}^{M} V_m v_m + \sum_{i=1}^{I} W_i w_i \quad \forall (\mathbf{u},\mathbf{v},\mathbf{w}) \in S \tag{8.16}$$

$$\mathbf{U},\mathbf{V},\mathbf{W} \geq 0, W_0 \in \mathbb{R}$$

Note that the affine approximation in (8.16) is equivalent to considering $2M+I$ basis function, each basis function defined as the value of one of the state variables.

The main advantages of using an affine value function approximation in the state variables are two. First, the approximate equivalent linear program (8.14) in Step 3 of the solution approach has only $2M+I+1$ variables, the number of approximation parameters, as opposed to $(C_r+1)^{M-1} \times (C_o+1)^{M-1} \times Q^I$, the number of cost-to-go values required to define the value function for each state of the system. Although (8.14) still has an intractable number of constraints, this allows us to solve its dual through column generation. Any other more sophisticated approximation architecture such as a linear combination of a set of more general basis functions would make the subproblem in the column generation algorithm a non-linear integer program. Second, the approximation parameters are directly interpretable. The values of $\{U_m\}_{m=1}^{M}$, $\{V_m\}_{m=1}^{M}$ and $\{W_i\}_{i=1}^{I}$ can be interpreted as the marginal expected discounted cost of having an additional regular-hour appointment slot occupied on day m (from today), the marginal expected discounted cost of having an additional overtime appointment slot booked on day m (from today), and the marginal expected discounted cost of having one more patient of type i waiting to be booked, respectively.

For Step 4, we use column generation to find the optimal solution to (8.17), the dual of the approximate equivalent linear program associated with the MDP model described above. Column generation finds the optimal solution to (8.17) starting with a small set of feasible state-action pairs and iteratively adding the state-action pair associated with the most violated primal constraint. Equations (8.18)–(8.20) and (8.21)–(8.23) are the constraint and objective function coefficients corresponding to the approximate equivalent linear program (8.14), respectively. They are obtained by substituting the affine approximation (8.16) into the equivalent linear program corresponding to (8.12), restricting γ to be an exogenous probability distribution over the initial state of the system, and rearranging terms.

$$\min_{\mathbf{X} \geq \mathbf{0}} \left\{ \sum_{\mathbf{s} \in S} \sum_{\mathbf{a} \in A_\mathbf{s}} c(\mathbf{s},\mathbf{a}) X(\mathbf{s},\mathbf{a}) \right\} \tag{8.17}$$

subject to

$$(1-\alpha) \sum_{\mathbf{s} \in S} \sum_{\mathbf{a} \in A_\mathbf{s}} X(\mathbf{s},\mathbf{a}) = 1$$

$$\sum_{\mathbf{s} \in S} \sum_{\mathbf{a} \in A_\mathbf{s}} \mu_m(\mathbf{s},\mathbf{a}) X(\mathbf{s},\mathbf{a}) \geq \mathbb{E}_\gamma[u_m] \qquad \forall m$$

$$\sum_{\mathbf{s}\in S}\sum_{\mathbf{a}\in A_\mathbf{s}} v_m(\mathbf{s},\mathbf{a})X(\mathbf{s},\mathbf{a}) \geq \mathbb{E}_\gamma[v_m] \qquad \forall m$$
$$\sum_{\mathbf{s}\in S}\sum_{\mathbf{a}\in A_\mathbf{s}} \omega_i(\mathbf{s},\mathbf{a})X(\mathbf{s},\mathbf{a}) \geq \mathbb{E}_\gamma[w_i] \qquad \forall i$$

where

$$\mu_m(\mathbf{s},\mathbf{a}) = u_m(\mathbf{s}) - \alpha u'_m(\mathbf{s},\mathbf{a}) \qquad \forall m, \mathbf{a}\in A_\mathbf{s}, \mathbf{s}\in S \tag{8.18}$$
$$v_m(\mathbf{s},\mathbf{a}) = v_m(\mathbf{s}) - \alpha v'_m(\mathbf{s},\mathbf{a}) \qquad \forall m, \mathbf{a}\in A_\mathbf{s}, \mathbf{s}\in S \tag{8.19}$$
$$\omega_i(\mathbf{s},\mathbf{a}) = w_i(\mathbf{s}) - \alpha\left(w_i(\mathbf{s}) - \sum_{n=1}^{N} x_{in} + m_i\right) \qquad \forall i, \mathbf{a}\in A_\mathbf{s}, \mathbf{s}\in S \tag{8.20}$$
$$\mathbb{E}_\gamma[u_m] = \sum_{\mathbf{s}\in S}\gamma(\mathbf{s})u_m(\mathbf{s}) \qquad \forall m \tag{8.21}$$
$$\mathbb{E}_\gamma[v_m] = \sum_{\mathbf{s}\in S}\gamma(\mathbf{s})v_m(\mathbf{s}) \qquad \forall m \tag{8.22}$$
$$\mathbb{E}_\gamma[w_i] = \sum_{\mathbf{s}\in S}\gamma(\mathbf{s})w_i(\mathbf{s}) \qquad \forall i \tag{8.23}$$

The model used to identify the state-action pair associated with the most violated primal constraint, the pricing problem, is itself an optimization model. Given the dual values associated with the current solution to (8.17), $\{U_m\}_{m=1}^M$, $\{V_m\}_{m=1}^M$ and $\{W_i\}_{i=0}^I$, the next state-action pair to enter the formulation is given by:

$$\underset{\mathbf{s}\in S,\mathbf{a}\in A_\mathbf{s}}{\arg\min}\left\{c(\mathbf{s},\mathbf{a}) - (1-\alpha)W_0 - \sum_{m=1}^{M}\mu_m(\mathbf{s},\mathbf{a})U_m - \sum_{m=1}^{M}v_m(\mathbf{s},\mathbf{a})V_m - \sum_{i=1}^{I}\omega_i(\mathbf{s},\mathbf{a})W_i\right\}$$

Column generation iterates until no primal constraint is violated, giving us $\{U_m^*\}_{m=1}^M$, $\{V_m^*\}_{m=1}^M$ and $\{W_i^*\}_{i=0}^I$, the optimal values of the approximation parameters. It can be shown that, under some necessary conditions regarding the expected demand over the infinite horizon and the expected number of appointment slots initially filled, the analytical solution to (8.17) is given by the following equations[2]:

$$U_m^* = \begin{cases} U_{m+l_1}^* & m = 1,\ldots,(T_1-1) \\ \alpha^{(m-1)}h & m = T_1,\ldots,(M-1) \\ 0 & m = M \end{cases} \tag{8.24}$$
$$V_m^* = 0 \qquad \forall m \tag{8.25}$$
$$W_i^* = \sum_{k=T_i}^{T_i+l_i-1} r_{i(k+1-T_i)}U_k^* \qquad \forall i \tag{8.26}$$

[2] A proof of the form of the optimal affine value function approximation for a simpler version of the advance patient appointment scheduling problem studied in this chapter can be found in [21].

In Eqs. (8.24) and (8.26), l_1 and T_1 represent the time span and the wait time target for patients with the highest priority. The optimal form of W_0^* is not provided since it does not play a role in solving (8.15), the minimization problem used to identify the approximate optimal actions in Step 5 of the solution approach.

When the necessary conditions are violated, the optimal values of the approximation parameters are zero and the optimal scheduling policy becomes a myopic policy. This, for example, happens when capacity is not a significant limitation.

8.5 Practical Results

To illustrate the quality of the appointment scheduling policies obtained through the solution approach described above, we review two practical applications found in the literature. We investigate the performance and practical implications of the resulting appointment scheduling policies using simulation, and compare their performance to that of myopic policies. Myopic policies are usually representative of current practices. They considers only what is known at the present (the immediate cost), ignoring the future impact of today's decisions.

8.5.1 Computerized Tomography Scan Appointment Scheduling

We first consider the dynamic multi-priority patient scheduling problem for a diagnostic resource in [21]. Using the methodological approach described above, the authors formulate and solve an advance patient appointment scheduling problem in which all patients require only one appointment slot and booking overtime in advance of the service day is not allowed.[3] The authors define c_{in}, the penalty associated with booking the appointment of a patient of type i on day n (from today), as a function of g_i, the penalty associated with delaying the scheduling decisions of a patient of type i, and α, the discount factor. In this way, the cost of delaying a patient's scheduling decisions k days and then booking his/her appointment within the wait time target is equal to the cost of booking the patient's appointment k days late initially. In their setting, c_{in} is non-increasing in i (i.e., patients with a smaller index i have a higher priority).

$$c_{in} = \begin{cases} 0, & n \leq T_i; \\ \sum_{k=T_i}^{n} \alpha^{k-T_i-1} g_i, & n > T_i. \end{cases} \qquad \forall i, n \tag{8.27}$$

[3] This is equivalent to assuming $l_i = 1\ \forall i$, $r_{i1} = 1\ \forall i$ and $y_m = 0\ \forall m \neq 1$.

Using the optimal values of the approximation parameters, the authors derive a scheduling policy that applied to the problem setting described later in this section translates into the following booking guidelines:

- Book as much type 1 demand as possible into the day interval $[1, T_1]$, starting with day 1 and working up to day T_1;
- For each successive patient type i, book incoming requests into any available appointment slot in the day interval $[1, T_i]$ starting with day 1, then day T_i, and working down to day 2;
- Use overtime for any remaining requests.

Figure 8.3 shows the performance of the resulting booking guidelines and that of a myopic policy for a small clinic with regular-hour capacity of 10 appointment slots per day. The myopic policy in this case books appointment requests as soon as possible, in decreasing priority order, according to the immediate cost function defined in (8.9). It resorts to overtime for patients of type i only when there is no available regular-hour capacity within the first $\bar{n}_i$ days of the booking horizon, where $\bar{n}_i = \max\{n : c_{in} < h\}$. The clinic divides demand into three priority classes with wait time targets of 7, 14 and 21 days, and it chooses a 21-day booking horizon. Demand from each patient type is assumed to be Poisson with means of 5, 3 and 2 appointment requests per day, respectively.[4] The overtime cost is 100 per appointment slot, the postponement penalties are 20, 10 and 5, respectively, and the discount factor is 0.99. There is no limit on the use of overtime. The information defining the problem setting is summarized in Table 8.1.

Table 8.1: Problem setting for a small clinic example

System information	
Regular-hour capacity (C_r)	10 slots/day
Booking horizon (M)	21 days
Overtime cost (h)	\$100
Discount factor (α)	0.99

Patient information	Type 1	Type 2	Type 3	
Wait time targets (T_i)	7	14	21	days
Demand rates (m_i)	5	3	2	requests/day
Postponement penalties (g_i)	\$20	\$10	\$5	

[4] Poisson distributions are truncated at three times their mean values to maintain a finite state space.

Each policy was simulated for 1600 days using common patient arrivals with statistics collected for each of 1000 simulation runs after a warm-up period of 200 days. The warm-up period was simulated under the resulting booking guidelines. The simulation results presented in Fig. 8.3 are summarized in Table 8.2.

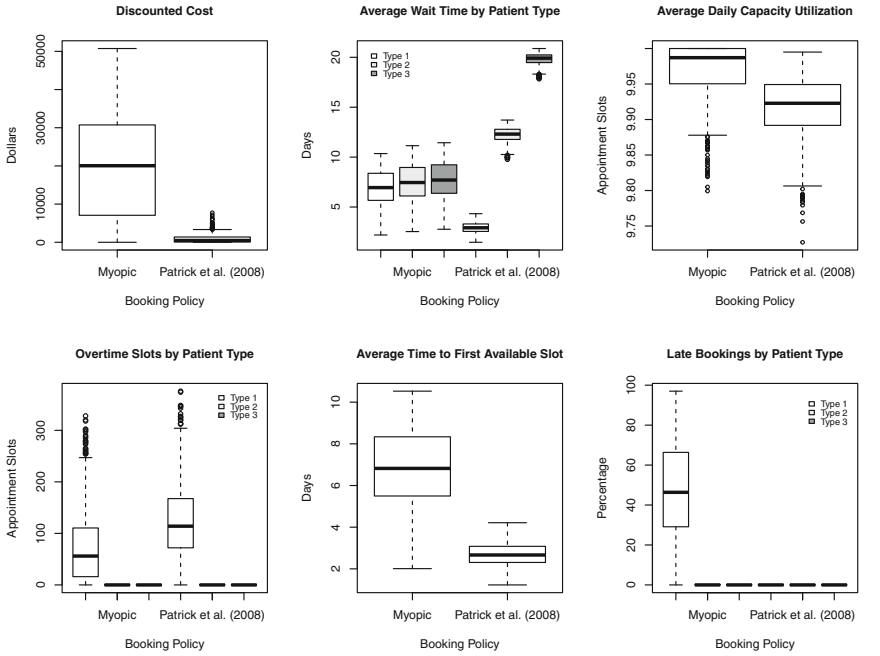


Fig. 8.3: Performance of the scheduling policies in terms of different metrics for a small clinic with a regular-hour capacity of 10 appointment slots per day. Each side-by-side box-plot was constructed with the results of 1000 1600-day simulation runs using common patient arrivals with statistics collected after a warm-up period of 200 days

The resulting booking guidelines clearly outperform the myopic policy in terms of the mean discounted cost. The myopic policy is better than the resulting scheduling policy with respect to the mean average wait time for lower priority patients, the mean average daily capacity utilization, and the mean overtime utilization. However, it is much worse in terms of the mean average wait time for the highest priority patients, the mean average time to the first available appointment slot, and the mean percentage of late bookings. It books, on average, almost 24 % of the patients late compared to 0 % for the resulting booking guidelines. The resulting scheduling policy still uses overtime, on average, for about 2 % of the highest priority patients.

Table 8.2: Comparison of the performance of the scheduling policies for a small clinic with a regular-hour capacity of 10 appointment slots per day. The corresponding 95 % confidence interval is provided for each statistic

Performance Metric	Patient Type	Myopic	Patrick et al. [21]
Discounted cost	–	19,507 ± 813	919 ± 70
Avg. wait time	1	6.95 ± 0.11	2.93 ± 0.03
	2	7.49 ± 0.12	12.24 ± 0.05
	3	7.74 ± 0.12	19.83 ± 0.03
Avg. daily capacity utilization	–	9.97 ± 0.00	9.92 ± 0.00
Overtime	1	73.17 ± 4.26	123.56 ± 4.33
	2	0.00 ± 0.00	0.00 ± 0.00
	3	0.00 ± 0.00	0.00 ± 0.00
Avg. time to first available slot	–	6.86 ± 0.11	2.70 ± 0.03
Percentage late	1	47.55 ± 1.49	0.00 ± 0.00
	2	0.00 ± 0.00	0.00 ± 0.00
	3	0.00 ± 0.00	0.00 ± 0.00

8.5.2 Radiation Therapy Treatment Scheduling

We now consider the radiation therapy treatment scheduling problem in [24]. Using the methodological approach described in this chapter, the authors formulate and solve an advance patient appointment scheduling problem that, in addition to multiple priority levels, considers patients who require appointments across multiple days and for irregular lengths of time. Radiation therapy treatments are typically classified into types according to cancer site, intent and urgency level. For most types of cancer, treatments are delivered in daily consecutive sessions for a time period that may vary between 1 day and 8 weeks, with breaks on weekends, and in which each session may last from 12 to 60 min. The authors also allow the possibility of scheduling overtime on different days of the booking horizon, and not necessarily for entire service sessions. The only difference with respect to the problem setting described in Sect. 8.2 is that the authors do not allow the possibility of postponing to the next day the scheduling decisions for some patients.[5]

The authors define c_{in}, the penalty (if any) for starting a treatment of type i on day n (from today), as the sum of discounted penalties f_{ik} associated with each additional day of wait before the start of a treatment. That is $c_{in} = \sum_{k=1}^{n} \alpha^{k-1} f_{ik} \ \forall i, n$. The values of f_{ik} are specified in relation to existing guidelines for acceptable waits and taking into consideration the importance of radiation therapy for different disease

[5] This is equivalent to imposing $\sum_{n=1}^{N} x_{in} = w_i \ \forall i$ in the definition of the action sets.

sites. Using the optimal values of the approximation parameters, the authors derive a scheduling policy which they evaluate via simulation for a practical example based on data provided by the British Columbia Cancer Agency (BCCA).

The authors compare the service levels of the resulting scheduling policy with those of a myopic policy for a cancer centre with regular-hour capacity of 120 appointment slots per day, which is equivalent to three identical treatment units operating eight hours a day (appointment slots are 12 min long). The centre divides demand into 18 treatment types and chooses a 100-day booking horizon. Demand for treatment is assumed Poisson with mean 8.25 requests per day.[6] The treatment types and the arrival process are described in Table 8.3. Treatments of type 1, for example, represent urgent lung, prostate and breast palliative cases and consist of an initial session of two appointment slots (i.e., 1×2) plus four additional sessions of one appointment slot each (i.e., 4×1). In other words, $\mathbf{r}_1 = (2,1,1,1,1)$. Thus, treatments of type 1 require 5 sessions and a total of 6 appointment slots. Their arrival rate is 0.19 requests per day. The overtime capacity is 15 appointment slots (1 h per treatment unit), the overtime cost is 100 per appointment slot, and the discount factor is 0.99. The daily wait time penalties are defined in Table 8.4.

Each policy was simulated for 1500 days using common patient arrivals with statistics collected for each of 10 simulation runs after a warm-up period of 750 days. Initial states were generated randomly. The simulation results are summarized in Table 8.5.

The resulting scheduling policy clearly outperforms the myopic policy with respect to the total number of cases initiated within 1, 5 and 10 workdays. The average percentage of treatments initiated within 1, 5 and 10 workdays increases from 5 % to 26 %, 29 % to 53 % and 73 % to 96 %, respectively. This is achieved at the expense of a negligible but statistically significant increase in the average overtime utilization of 3 min a day. Similar improvements in these service levels are observed for most treatment types individually.

The main drawback of the resulting scheduling policy is the increase in the wait times for treatments of types 1 to 3. The fact that the most time-sensitive treatments wait slightly longer, however, demonstrates a willingness to trade-off a small increase in wait time for more time-sensitive treatments for a larger gain for less time-sensitive treatments. The simulation results also show no statistically significant difference between the two policies in terms of the average daily regular-hour capacity utilization. Both policies use on average about 99.5 % of the available regular-hour capacity. Still, the resulting scheduling policy outperforms the myopic policy with respect to the average discounted cost. The average discounted cost associated with the resulting scheduling policy is \$121,974 while the average discounted cost associated with the myopic policy is \$185,843.

[6] A maximum number of requests, obtained from historical data, is set to maintain a finite state space.

Table 8.3: Characteristics of the problem setting used to evaluate the performance of the resulting treatment scheduling policy

System information

Regular-hour capacity (C_r)	120 slots/day
Overtime capacity (C_o)	15 slots/day
Booking horizon (M)	136 days
Overtime cost (h)	\$100
Discount factor (α)	0.99

Treatment information

Treatment type i	Capacity requirements (r_{ij})	Sessions/ slots	Arrival rate (m_i) [reqs./day]	Most frequent case (site and intent)
1	1×2 + 4×1	5/ 6	0.19	Lung/Prostate/Breast
2	1×2	1/ 2	0.11	Palliative
3	1×2 + 3×1	4/ 5	0.11	(Urgent)
4	1×2 + 15×1	16/17	1.43	Breast Adjuvant
5	1×2 + 15×1 + 1×2 + 3×1	20/22	0.59	
6	1×3 + 15×2	16/33	0.45	
7	1×2	1/ 2	1.42	Lung/Breast/Prostate
8	1×2 + 4×1	5/ 6	1.36	Palliative
9	1×2 + 9×1	10/11	0.57	(Non-urgent)
10	1×2 + 3×1	4/ 5	0.38	
11	1×2 + 14×1	15/16	0.18	
12	1×1	1/ 1	0.18	
13	1×2 + 19×1	20/21	0.29	Head and Neck Radical
14	1×3 + 34×2	35/71	0.21	
15	1×2 + 32×1	33/34	0.30	Prostate Radical
16	1×2 + 36×1	37/38	0.29	
17	1×2 + 21×1 + 1×2 + 14×1	37/39	0.15	
18	1×2 + 32×1	33/34	0.04	Prostate Adjuvant

8.6 Discussion

The results for the two practical applications above suggest that the methodological approach described in this chapter provides a systematic way of identifying effective decision policies for advance patient appointment scheduling problems. By adequately assessing the future impact of today's decisions, appointment scheduling policies derived using this approach, although possibly sub-optimal, perform significantly better than myopic policies which are the most common representation of current scheduling practices. They provide clear guidelines as to when to book patients of each type and as to when to resort to overtime or to any other alternative source of surge capacity. These policies usually book lower priority patients further

Table 8.4: Daily wait time penalties associated with each treatment type. The values of the wait time penalties were determined by expert opinion and investigated through sensitivity analysis

Types	Daily penalty within workday interval						
	[0,1]	(1,5]	(5,10]	(10,20]	(20,30]	(30,40]	(40,100]
1–3	0	100	150	150	150	150	150
4–6	0	0	0	50	100	100	150
7–12	0	0	65	100	100	100	150
13–14	0	0	80	150	150	150	150
15–17	0	0	0	40	80	100	150
18	0	0	0	50	90	100	150

Table 8.5: Simulation results. The bold font indicates the policy that provides the highest service level for each treatment type and wait time target. Only the results for which a significance test shows the mean service level has improved (at 0.05 significance level) are highlighted. Note that no policy is highlighted for wait time targets of 15 and 20 days as both policies perform equally well (statistically speaking)

Type	Myopic Policy % of the cases initiated within					Sauré et al. (2012) % of the cases initiated within				
	1 day	5 days	10 days	15 days	20 days	1 day	5 days	10 days	15 days	20 days
1	70 ± 10	**94 ± 4**	**100 ± 0**	100 ± 0	100 ± 0	66 ± 7	81 ± 8	97 ± 2	100 ± 0	100 ± 0
2	**82 ± 8**	**95 ± 5**	**100 ± 0**	100 ± 0	100 ± 0	75 ± 9	84 ± 8	97 ± 2	100 ± 0	100 ± 0
3	72 ± 12	**95 ± 5**	**100 ± 0**	100 ± 0	100 ± 0	66 ± 11	80 ± 10	97 ± 3	100 ± 0	100 ± 0
4	1 ± 2	4 ± 3	57 ± 9	98 ± 2	100 ± 0	**17 ± 6**	**17 ± 6**	**95 ± 4**	100 ± 0	100 ± 0
5	1 ± 1	4 ± 3	55 ± 10	98 ± 2	100 ± 0	**11 ± 5**	**11 ± 5**	**94 ± 5**	100 ± 0	100 ± 0
6	1 ± 1	4 ± 3	53 ± 10	97 ± 3	100 ± 0	**12 ± 5**	**12 ± 5**	**94 ± 5**	100 ± 0	100 ± 0
7	2 ± 2	75 ± 4	**100 ± 0**	100 ± 0	100 ± 0	43 ± 9	82 ± 8	97 ± 3	100 ± 0	100 ± 0
8	2 ± 2	25 ± 9	76 ± 5	100 ± 1	100 ± 0	**31 ± 9**	**79 ± 9**	**96 ± 3**	100 ± 0	100 ± 0
9	2 ± 2	16 ± 10	62 ± 9	99 ± 2	100 ± 0	**27 ± 8**	**78 ± 9**	**96 ± 3**	100 ± 0	100 ± 0
10	2 ± 2	37 ± 8	89 ± 2	100 ± 0	100 ± 0	**38 ± 8**	**81 ± 8**	**96 ± 3**	100 ± 0	100 ± 0
11	3 ± 2	15 ± 10	61 ± 10	99 ± 1	100 ± 0	**25 ± 9**	**77 ± 9**	**96 ± 3**	100 ± 0	100 ± 0
12	4 ± 2	92 ± 2	100 ± 0	100 ± 0	100 ± 0	**36 ± 7**	**97 ± 2**	100 ± 0	100 ± 0	100 ± 0
13	1 ± 1	19 ± 12	90 ± 3	100 ± 0	100 ± 0	**23 ± 9**	**80 ± 8**	**97 ± 3**	100 ± 0	100 ± 0
14	1 ± 1	16 ± 10	61 ± 9	99 ± 2	100 ± 0	**11 ± 4**	**77 ± 9**	**95 ± 3**	100 ± 0	100 ± 0
15	1 ± 1	**2 ± 2**	54 ± 9	97 ± 3	100 ± 0	0 ± 0	0 ± 0	**94 ± 5**	100 ± 0	100 ± 0
16	0 ± 0	**2 ± 1**	52 ± 10	97 ± 3	100 ± 0	0 ± 0	0 ± 0	**94 ± 4**	100 ± 0	100 ± 0
17	1 ± 1	2 ± 2	52 ± 8	97 ± 3	100 ± 0	0 ± 0	0 ± 0	**94 ± 5**	100 ± 0	100 ± 0
18	1 ± 1	**3 ± 3**	55 ± 11	98 ± 2	100 ± 0	0 ± 0	0 ± 0	**93 ± 6**	100 ± 0	100 ± 0
Total	5 ± 2	29 ± 4	73 ± 6	99 ± 1	100 ± 0	**26 ± 7**	**53 ± 6**	**96 ± 3**	100 ± 0	100 ± 0

into the future allowing the appointments for higher priority patients to be scheduled earlier. In this way, they achieve reasonable average wait times at a much lower discounted cost. Patient scheduling guidelines obtained through this approach could potentially be used in practice to increase service levels and thus to decrease the potential impact of waits on patients' health.

This chapter focuses on the use of linear programming and an affine value function approximation in the state variables as a means of solving advance patient appointment scheduling problems, which are problems computationally intractable using standard solution techniques. However, results obtained by Sauré et al. [25] suggest that the actual value function for the dynamic multi-priority patient scheduling problem in [21] can be better represented by a generalized logistic function in the state variables. The authors show that advance patient appointment scheduling policies obtained through a simulation-based algorithm using this type of non-linear approximation perform better, in most cases, than other booking policies for this problem. In particular, they provide lower discounted cost values and shorter average wait times for higher priority patients than policies derived directly using the methodological approach described in this chapter. In general, these policies book lower priority patients earlier, highest priority patients late, and use overtime preemptively, all depending on the level of congestion of the system.

Even though advance patient appointment scheduling policies obtained using a linear programming approach and an affine value function approximation in the state variables may not be as good as policies obtained from a simulation-based method using a parametric non-linear value function approximation, the additional effort required to solve the latter makes the former a better choice for large scale problems. The slow speed of simulation and the complexity of the optimization problem solved at each decision epoch (a non-linear integer program) would make the solution times quickly intractable for large scale problems. In this sense, although the linear programming algorithm described in this chapter is limited to affine approximation architectures in the state variables, its main advantage is that it avoids the need for iterative learning and often provides good results.

We believe that the methodological approach described in this chapter can be adapted to any dynamic problem that involves allocating a fixed amount of daily capacity among entities of different types. For example, to order acceptance problems and to more general capacity allocation problems involving multiple demand origins and multiple service providers. However, its use requires the immediate cost and the value function to be represented as a linear combination of individual state components (e.g., the number of entities at each demand origin, available capacity at each service provider, processing capacity on each calendar day, location of a given resource, etc.). A simulation-based method, given its problem-size limitations, could be used to solve small examples in order to gain a better understanding of the main characteristics of more complex policies for a given problem. Any additional insights could be later tested through simulation.

8.7 Open Challenges

There are numerous potential extensions to the methodological approach presented in this chapter. The main two involve elements of the advance patient appointment scheduling problem that were excluded from its mathematical formulation. The model assumes that the portion of the demand that is not booked today (i.e., the number of patients whose scheduling decisions are postponed to the next day) is handled the same as the new demand tomorrow. This is done in order to avoid the additional complexity of keeping track of patient wait times. In addition, the model implicitly assumes that the regular-hour capacity and the overtime capacity are determined by aggregating individual capacities from multiple identical resources. This assumption may be unrealistic for some settings in which medical resources differ in terms of their technical capabilities and, consequently, in which different types of patients need to be assigned to specific groups of resources. The inclusion of wait lists and multiple resources, together with patient-resource compatibility constraints, would clearly make a solution to the model more challenging but, at the same time, it would increase the range of systems to which the solution approach could be applied.

We have formulated and approximately solved a simple model that explicitly considers wait lists by keeping track of the number of patients of each type who have waited a given number of days without receiving an appointment [23]. Preliminary results, in absence of an immediate postponement cost, indicate that the resulting scheduling policies would make patients wait as long as possible before booking them in decreasing priority order into the first available appointment slot. This is equivalent to defining demand as the number of patients whose booking decisions cannot be postponed any longer and using a first available appointment slot policy.

In addition, the methodological approach described in this chapter assumes that service times are deterministic. Consequently, overtime simply involves subtracting the regular-hour capacity from the capacity requirements associated with the number of appointment booked on a given day, and taking the positive part. Incorporating stochastic services times would make the solution approach much more complicated as the sequence and the spacing of the appointments on the service day may have a major impact on the realized overtime. The allocation schedule (the sequence of patients and their respective appointment times) would have a clear impact on the optimal advance schedule (the number and type of patients to book on each day), and conversely, the optimal allocation schedule would depend on the advance schedule as one of its inputs is the number of patients of each type booked on each day. To date, the allocation scheduling and the advance scheduling problems have been solved separately. However, there are numerous research initiatives that seek to bridge these two problems by, for example, incorporating appointment sequencing decisions into the MDP model in [21] and using the algorithm developed by Begen and Queyranne [3] to determine the overtime associated with choosing optimal patient appointment times [4].

Finally, it is important to note that the solution approach presented in this chapter has been developed as a proof of concept. Its application to problems faced by particular medical facilities would require further research but it would certainly

be plausible. Our work to date has focused on the methodological framework for how to address advance patient appointment scheduling problems rather than on the actual implementation of the resulting policies. In any case, we expect the main implementation challenges to be related to database connectivity, user interface and software functionalities, in addition to some possible cultural reluctance, rather than to the actual scheduling program (a simple integer program or application reflecting the main scheduling guidelines). It is also important to note that a second level of scheduling is needed to assign patients to specific appointment times.

Appendix: Notation

S	State space
$\mathbf{s}, \mathbf{s}'$	State vectors
$\mathbf{a}$	An action vector
$A_{\mathbf{s}}$	Set of actions available in state $\mathbf{s}$
$c(\mathbf{s}, \mathbf{a})$	One step cost in state $\mathbf{s}$, under action $\mathbf{a}$
α	Discount factor
δ	Decision rule
$\mathbf{s}_t$	State at time t
$\delta(\mathbf{s}_t)$	Action at time t
$\pi = (\delta, \delta, \dots)$	Stationary policy
$p(\mathbf{s}'\|\mathbf{s}, \mathbf{a})$	Transition probability into state $\mathbf{s}'$, when in state $\mathbf{s}$, under action $\mathbf{a}$
$v(\mathbf{s})$	Expected discounted cost over infinite horizon, starting in state $\mathbf{s}$
$\gamma(\mathbf{s})$	State relevance weight of state $\mathbf{s}$
$\phi_k(\mathbf{s})$	Basis function evaluated in state $\mathbf{s}$
U_m, V_m, W_i	Value function approximation parameters

References

1. D. Adelman, D. Klabjan, Computing near-optimal policies in generalized joint replenishment. INFORMS J. Comput. **24**(1), 148–164 (2012)
2. D. Adelman, A. Mersereau, Relaxations of weakly coupled stochastic dynamic programs. Oper. Res. **56**(3), 712–727 (2008)
3. M. Begen, M. Queyranne, Appointment scheduling with discrete random durations. Math. Oper. Res. **36**(2), 240–257 (2011)
4. M. Begen, J. Patrick, A. Sauré, Dynamic multi-priority, multi-class patient scheduling with stochastic service times. Working Paper (2016)
5. W. Ben-Ameur, J. Neto, Acceleration of cutting-plane and column generation algorithms: applications to network design. Networks **49**(1), 3–17 (2007)
6. D. Bertsekas, J. Tsitsiklis, *Neuro-Dynamic Programming* (Athena Scientific, Belmont, MA, 1996)

7. B. Cardoen, E. Demeulemeester, J. Belien, Operating room planning and scheduling: a literature review. Eur. J. Oper. Res. **201**(3), 921–932 (2010)
8. T. Cayirli, E. Veral, Outpatient scheduling in health care: A review of literature. Prod. Oper. Manag. **12**(4), 519–549 (2003)
9. D. de Farias, B. Van Roy, The linear programming approach to approximate dynamic programming. Oper. Res. **51**(6), 850–865 (2003)
10. D. de Farias, B. Van Roy, On constraint sampling in the linear programming approach to approximate dynamic programming. Math. Oper. Res. **29**(3), 462–478 (2004)
11. F. d'Epenoux, A probabilistic production and inventory problem. Manag. Sci. **10**(1), 98–108 (1963)
12. C. Derman, *Finite State Markovian Decision Processes* (Academic Press, Inc., Orlando, FL, 1970)
13. V. Desai, V. Farias, C. Moallemi, A smoothed approximate linear program. Adv. Neural Inf. Process. Syst. **22**, 459–467 (2009)
14. A. Erdelyi, H. Topaloglu, Computing protection level policies for dynamic capacity allocation problems by using stochastic approximation methods. IIE Trans. **41**, 498–510 (2009)
15. Y. Gocgun, M. Puterman, Dynamic scheduling with due dates and time windows: an application to chemotherapy patient appointment booking. Health Care Manag. Sci. **17**(1) ,60–76 (2014)
16. D. Gupta, B. Denton, Appointment scheduling in health care: challenges and opportunities. IIE Trans. **40**(9), 800–819 (2008)
17. L. Kallenberg, *Linear Programming and Finite Markovian Control Problems* (Mathematisch Centrum, Amsterdam, 1983)
18. M. Lübbecke, J. Desrosiers, Selected topics in column generation. Oper. Res. **53**(6), 1007–1023 (2005)
19. J. Magerlein, J. Martin, Surgical demand scheduling: a review. Health Serv. Res. **13**(4), 418–433 (1978)
20. S. Mondschein, G. Weintraub, Appointment policies in service operations: a critical analysis of the economic framework. Prod. Oper. Manag. **12**(2), 266–286 (2003)
21. J. Patrick, M. Puterman, M. Queyranne, Dynamic multipriority patient scheduling for a diagnostic resource. Oper. Res. **56**(6), 1507–1525 (2008)
22. W. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality* (Wiley-Interscience, Hoboken, NJ, 2011)
23. A. Sauré, *Approximate Dynamic Programming Methods for Advance Patient Scheduling*. PhD thesis, The University of British Columbia, 2012
24. A. Sauré, J. Patrick, S. Tyldesley, M. Puterman, Dynamic multi-appointment patient scheduling for radiation therapy. Eur. J. Oper. Res. **223**(2), 573–584 (2012)
25. A. Sauré, J. Patrick, M.L. Puterman, Simulation-based approximate policy iteration with generalized logistic functions. INFORMS J. Comput. **27**(3), 579–595 (2015)

26. H. Schütz, R. Kolisch, Approximate dynamic programming for capacity allocation in the service industry. Eur. J. Oper. Res. **218**(1), 239–250 (2012)
27. P. Schweitzer, A. Seidmann, Generalized polynomial approximations in Markovian decision processes. J. Math. Anal. Appl. **110**(2), 568–582 (1985)
28. R. Sutton, A. Barto, *Reinforcement Learning: An Introduction* (MIT Press, Cambridge, MA, 1998)

Chapter 9
Optimal Ambulance Dispatching

C.J. Jagtenberg, S. Bhulai and R.D. van der Mei

Abstract This chapter considers the ambulance dispatch problem, in which one must decide which ambulance to send to an incident in real time. In practice as well as in literature, it is commonly believed that the closest idle ambulance is the best choice. This chapter describes alternatives to the classical closest idle ambulance rule. Our first method is based on a Markov decision problem (MDP), which constitutes the first known MDP model for ambulance dispatching. Moreover, in the broader field of dynamic ambulance management, this is the first MDP that captures more than just the number of idle vehicles, while remaining computationally tractable for reasonably-sized ambulance fleets. We analyze the policy obtained from this MDP, and transform it to a heuristic for ambulance dispatching that can handle the real-time situation more accurately than our MDP states can describe. We evaluate our policies by simulating a realistic emergency medical services region in the Netherlands. For this region, we show that our heuristic reduces the fraction of late arrivals by 13% compared to the "closest idle" benchmark policy. This result sheds new light on the popular belief that deviating from the closest idle dispatch policy cannot greatly improve the objective.

C.J. Jagtenberg (✉)
Stochastics, CWI, Science Park 123, 1098 XG Amsterdam, The Netherlands
e-mail: jagtenbe@cwi.nl

S. Bhulai
Faculty of Sciences, VU University Amsterdam, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
e-mail: s.bhulai@vu.nl

R.D. van der Mei
Stochastics, CWI, Science Park 123, 1098 XG Amsterdam, The Netherlands
e-mail: mei@cwi.nl

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_9

9.1 Introduction

Emergency Medical Services (EMS) providers serve emergency calls while aiming to keep response times short. In particular, one issue that plays a central role is to maximize the fraction of incidents that are reached within a certain target time. Operations research and mathematical modeling can help to reach this goal.

9.1.1 Previous Work

A large number of models are available for ambulance planning. On one hand, there are models that deal with planning on a strategic level. Typically, such models determine the best locations for ambulance bases [6], and/or the number of vehicles that should be positioned at each base [7, 9]. Most of these solutions use mixed integer linear programming models to solve the problem. On the other hand, there is previous work on *operational* ambulance planning. This has attracted a wider range of solution methods, including Markov decision theory [1] and simulation-based optimization [4].

The variety of solution methods for operational ambulance planning might be due to the difficulty of the problem. In dynamic ambulance management, the point of issue is to make decisions based on real-time information on the state of all vehicles and incidents. This makes for a complex issue, and systems quickly become intractable when the number of vehicles grows. Especially in urban areas, the situation can be considered extremely difficult because multiple vehicles operate closely to one another and therefore cannot be treated independently.

Some of the papers on operational ambulance planning use Markov Decision Problems (MDPs), or a variant thereof, to model the problem. Typically, those models can be divided into two categories: the first incorporates very little information, for example, only the *number* of idle ambulances in [1]. In contrast, the second category models many aspects of the real-time situation, resulting in an extremely large state space. Therefore, the MDP's in the latter case are not solvable with classical methods, and authors resort to other solutions such as approximate dynamic programming (e.g., [12, 14]).

The vast majority of the papers on dynamic ambulance management have focused on how to redeploy idle vehicles, (e.g., [1, 12, 17]). Perhaps in order not to overcomplicate things, they assume a basic dispatch rule: whenever an incident occurs, they decide to send the ambulance that is closest to the incident (in time). Although this is a common dispatch policy, it was already shown to be suboptimal in 1972 [5]. Regardless, most authors make this assumption without much discussion or justification; for example, the authors of [12] claim that it is an accurate enough representation of reality; however, they do not address the question of whether it is an *optimal* choice with respect to the objective (which is the fraction of incidents that is reached within the threshold time). The authors of [1] do not address the assumption at all.

Dispatching the closest idle ambulance seems to be so natural that justification is not needed, but one should not overlook the possibility to change the dispatch policy in order to improve the objective. In [17], the authors admit this is a possibility; however, they focus on relocating the vehicles when they become idle (instead of when they are dispatched). It should be clear that a clever dispatch policy can improve the performance of an EMS system, but since the topic has been underexposed in current literature, it is still unknown how much improvement can be expected. Furthermore, a dispatch method may be combined with a relocation rule to realize even greater improvements.

Few papers have discussed dispatch rules other than sending the closest idle ambulance. The ones that do propose other dispatch methods, typically do not prescribe a dynamic solution. For example, [4] proposes to divide the EMS region into separate sub-regions, each accompanied with its own ranking of bases from which an ambulance should preferably depart. Another example is [15], which considers a "regionalized response" dispatch policy. Under regionalized response, each region is preferably served by its own ambulance, even if it is temporarily outside of the region. Only if that vehicle is unavailable, the closest idle ambulance is sent. However, notice that both papers ignore information that we consider to be of crucial value: the outcome does not depend on whether some regions remain uncovered after the ambulance is dispatched. Alternatively, a choice could be made such that the remaining idle vehicles are in a good position with respect to future demand. This ensures that future incidents get a larger likelihood of being reached in time, thereby increasing the total expected fraction of incidents that can be reached within the time threshold.

One paper explicitly claims that alternative dispatch methods perform worse than the closest idle rule [8]. However, its true issue seems to be a computer-aided dispatch system that is not accurate enough to determine the true positions of the vehicles (and hence, also not able to determine the closest ambulance). The paper does not in fact deny that the "closest idle ambulance rule" might be improved. We emphasize that accurate location information is crucial in order to determine the best ambulance to send to an incident. Throughout this chapter, we will assume that such information is present. In many regions, such as Flevoland in the Netherlands, a monitoring tool is available that refreshes the exact GPS coordinates of each vehicle every 30 s. This seems accurate enough for our purposes.

9.1.2 Our Contribution

The main goal of this chapter is to better understand the ambulance dispatch process. In particular, we question the often-made assumption that one cannot do much better than the "closest idle" dispatch method. Thereto, we search for sensible dispatch rules other than the classical closest idle ambulance policy. We mainly focus on the often-used objective to minimize the fraction of arrivals later than a certain target time. However, we show that one of our methods can also be used for other KPI's.

First, we propose an MDP for ambulance dispatching, where the state space is described by an optional incident location and the availability of each of the ambulances. To the best of our knowledge, this is the first MDP in ambulance literature that models more than just the number of idle vehicles, without losing tractability for reasonably-sized ambulance fleets. In some sense, this model balances the amount of detail in the system representation—which typically results in a better outcome—with the computational difficulties. We mainly focus on minimizing the fraction of arrivals later than a target time, a typical objective in ambulance planning. However, we show that with a small change, our model can also minimize the average response time.

Second, we propose a heuristic for ambulance dispatching that behaves similar to the policy obtained from the MDP. However, it is able to determine more accurately what the response time would be when dispatching a driving ambulance. Furthermore, the heuristic can be computed in polynomial time, which allows us to apply it to regions with a large number of vehicles.

We validate our policies by a discrete-event simulation model of a Dutch EMS region. These simulations indicate that our proposed dispatch heuristic can decrease the fraction of late arrivals by as much as 13% relatively compared to the closest idle ambulance dispatch method. Our result sheds new light on the popular belief that deviating from the closest idle policy cannot greatly improve the objective. Although we do not advise all EMS managers to immediately discard the closest idle dispatch method, we do show that the typical argument—that it would not lead to large improvements in the fraction of late arrivals—should be changed.

The rest of this chapter is structured as follows. In Sect. 9.2, we give a formal problem definition. In Sect. 9.3, we present our proposed solution using Markov Decision Processes (MDPs), followed by a solution based on a scalable heuristic in Sect. 9.4. We show our results for a small, intuitive region in Sect. 9.5 and in a realistic case study for the Dutch area of Flevoland in Sect. 9.6.

9.2 Problem Formulation

Define the set V as the set of locations at which incidents can occur. Note that these demand locations are modeled as a set of discrete points. Incidents at locations in V occur according to a Poisson process with rate λ. Let d_i be the fraction of the demand rate λ that occurs at node i, $i \in V$. Then, on a smaller scale, incidents occur at node i with rate λd_i.

Let A be the set of ambulances, and $A_{idle} \subseteq A$ the set of currently idle ambulances. When an incident has occurred, we require an idle ambulance to immediately drive to the scene of the incident. The decision which ambulance to send has to be made at the moment we learn about the incident, and is the main question of interest in this chapter. When an incident occurs and there are no idle ambulances, the call goes to a first-come first-serve queue.

V	The set of demand locations.
H	The set of hospital locations, $H \subseteq V$.
A	The set of ambulances.
A_{idle}	The set of idle ambulances.
W_a	The base location for ambulance a, $a \in A$, $W_a \in V$.
T	The time threshold.
λ	Incident rate.
d_i	The fraction of demand in i, $i \in V$.
$\tau_{i,j}$	The driving time between i and j with siren turned on, $i, j \in V$.

Table 9.1: Notation

Our objectives are formulated in terms of response times: the time between an incident and the arrival of an ambulance. In practice, incidents have the requirement that an ambulance must be present within T time units. Therefore, we want to minimize the fraction of incidents for which the response time is larger than T. Another observation is that we want response times to be short, regardless of whether they are smaller or greater than T. We translate this into a separate objective, which is to minimize the average response time. We assume that the travel time $\tau_{i,j}$ between two nodes $i, j \in V$ is deterministic, and known in advance.

Sending an ambulance to an incident is followed by a chain of events, most of which are random. When an ambulance arrives at the incident scene, it provides service for a certain random time τ_{on_scene}. Then it is decided whether the patient needs transport to a hospital. If not, the ambulance immediately becomes idle. Otherwise, the ambulance drives to the nearest hospital in the set $H \subseteq V$. Upon arrival, the patient is transferred to the emergency department, taking a random time $\tau_{hospital}$, after which the ambulance becomes idle.

An ambulance that becomes idle may be dispatched to another incident immediately. Alternatively, it may return to its base location. Throughout this chapter, we will assume that we are dealing with a *static* ambulance system, i.e., each ambulance has a fixed, given base and may not drive to a different base. However, it is possible that multiple ambulances have the same base location. We denote the base location of ambulance a by W_a, for $a \in A$.

An overview of the notation can be found in Table 9.1.

9.3 Solution Method: Markov Decision Process

We model the ambulance dispatch problem as a discrete-time Markov Decision Process (MDP). In each state s (further defined in Sect. 9.3.1), we must choose an action from the set of allowed actions: $\mathcal{A}_s \subseteq \mathcal{A}$, which we describe in Sect. 9.3.2. The process evolves in time according to transition probabilities that depend on the chosen actions, as described in Sect. 9.3.4. We are dealing with an infinite planning

horizon, and our goal is to maximize the average reward. The rewards are defined in Sect. 9.3.3. We eventually find our solution by performing value iteration [13].

In our model, we assume that at most one incident occurs within a time step. Therefore, the smaller the time steps, the more accurate the model will be. However, there is a tradeoff, as small time steps will increase the computation time. Throughout this chapter, we take time steps to be 1 min, which balances the accuracy and the computation time.

9.3.1 State Space

When designing a state space, it is important to store the most crucial information from the system in the states. However, when dealing with complex problems—such as real-time ambulance planning—it is tempting to store so much information, that the state space becomes intractable. This would lead to the so-called curse of dimensionality [2], which makes it impossible to solve the problem with well-known Markov Decision Problem (MDP) approaches.

As discussed before, there is little previous work on how to choose a good dispatch policy, but to some extent we can draw parallels with work on dynamic ambulance redeployment (which relocates idle vehicles): some researchers overcome the problem of an intractable state space by turning to Approximate Dynamic Programming, which allows for an elaborate state space to be solved approximately [12]. Alternatively, some researchers choose a rather limited state space, for example, by describing a state merely by the *number* of idle vehicles [1].

For our purpose, i.e., to determine *which* ambulance to send, it is important to know whether the ambulance we might send will arrive within T time units. Therefore, it is crucial to know where the incident took place. Furthermore, we require some knowledge of where the idle ambulances are. Clearly, storing only the number of idle vehicles would be insufficient. However, storing the location of each idle ambulance would already lead to an intractable state space for practical purposes. Instead, we can benefit from the fact that we are trying to improve a *static* solution. In a static solution, the home base for any ambulance is known in advance. Note that an idle ambulance must be either residing at its base location, or traveling towards the base. Hence, if we allow for an inaccuracy in the location of idle ambulances, in the sense that we use their destination rather than their actual location, their location does not need to be part of the state. Merely keeping track of whether each ambulance is idle or not, now suffices.

This leads us to a state s, defined as follows.

$$(Loc_{acc}, idle_1, idle_2, \ldots, idle_{|A|}), \tag{9.1}$$

where Loc_{acc} denotes the location of the incident that has just occurred in the last time step. In case no incident occurred in the last time step, we denote this by a dummy location, hence

$$Loc_{acc} \in V \cup \{0\}.$$

Furthermore, $idle_i$ denotes whether ambulance i is idle:

$$idle_i \in \{True, False\}, \quad \forall i \in A.$$

This leads to a state space of size $(|V|+1)2^{|A|}$.

For future reference, let $Loc_{acc}(s)$ denote the location of the incidents that have occurred in the previous time step when the system is in state s. Let $idle_i(s)$ denote whether or not ambulance i is idle in state s, $\forall i \in A, \forall s \in S$.

9.3.2 Policy Definition

In general, a policy Π can be defined as a mapping from the set of states to a set of actions: $S \to \mathcal{A}$. In our specific case, we define $\mathcal{A} = A \cup \{0\}$; that is if $\Pi(s) = a$, for $a \in A$, ambulance a should be sent to the incident that has just occurred at $Loc_{acc}(s)$. Action 0 may be interpreted as sending no ambulance at all (this is typically the choice when no incident occurred in the last time step, or when no ambulance is available).

In a certain state, not all actions are necessarily allowed. Denote the set of feasible actions in state s as

$$\mathcal{A}_s \subseteq \mathcal{A}, \quad \forall s \in S.$$

For example, it is not possible to send an ambulance that is already busy with another incident. This implies

$$!idle_a(s) \to a \notin \mathcal{A}_s, \quad \forall a \in A, \quad \forall s \in S. \tag{9.2}$$

Furthermore, let us require that when an incident has taken place, we must always send an ambulance—if any are idle.

$$\exists a \in A : idle_a(s) \ \wedge \ Loc_{acc}(s) \neq 0 \to 0 \notin \mathcal{A}_s, \quad \forall s \in S. \tag{9.3}$$

Moreover, if no incident has occurred, we may simplify our MDP by requiring that we do not send an ambulance:

$$Loc_{acc}(s) = 0 \to \mathcal{A}_s = \{0\}, \quad \forall s \in S. \tag{9.4}$$

All other actions from $\mathcal{A}$ that are not restricted by (9.2)–(9.4) are feasible. This completely defines the allowed action space for each state.

9.3.3 Rewards

In ambulance planning practice, a typical goal is to minimize the fraction of late arrivals. Since our decisions have no influence on the number of incidents that occur, this is equivalent to minimizing the *number* of late arrivals. An alternative goal might be to minimize average response times. Our MDP approach may serve either of these objectives, simply by changing the reward function.

Define $R(s,a)$ as the reward received when choosing action a in state s, $\forall s \in S, \forall a \in \mathcal{A}_s$. Note that in this definition, the reward does not depend on the next state. Keep in mind that our goal is to maximize the average rewards.

9.3.3.1 Fraction of Late Arrivals

To minimize the fraction of late arrivals, i.e., the fraction of incidents for which the response time is greater than T, we define the following rewards:

$$R(s,a) = \begin{cases} 0 & \text{if } Loc_{acc}(s) = 0; \\ -N & \text{if } Loc_{acc}(s) \neq 0 \ \wedge \ a = 0, \text{ i.e., no idle ambulances;} \\ 0 & \text{if } Loc_{acc}(s) \neq 0 \ \wedge \ a \in A \ \wedge \ \tau_{W_a,\, Loc_{acc}(s)} \leq T; \\ -1 & \text{otherwise.} \end{cases}$$

Here N is a number that is typically greater than 1. This implies that when all ambulances are busy, the rewards are smaller than when we send an ambulance that takes longer than T to arrive. This is in agreement with the general idea that having no ambulances available is a very bad situation. One might be tempted to make the reward for the only possible action ($a = 0$) in these states even smaller than we did, in order to influence the optimal actions in other states: the purpose would be to steer the process away from states with no ambulances available. However, note that this would not be useful, because our actions do not affect how often we end up in a state where all ambulances are busy. This is merely determined by the outcome of an external process, i.e., an unfortunate sequence of incidents. Therefore, an extremely small reward for action $a = 0$ in states where all ambulances are busy, would only blur the differences between rewards for actions in other states. (In our numerical experiments, we use $N = 5$.)

9.3.3.2 Average Response Time

To minimize the average response time, one may use the same MDP model, except with a different reward function. Let M be a large enough number, typically such that $M > \tau_{i,j}, \forall i, j \in V$. Then we can define the rewards as follows.

$$R(s,a) = \begin{cases} 0 & \text{if } Loc_{acc}(s) = 0; \\ -M & \text{if } Loc_{acc}(s) \neq 0 \ \wedge \ a = 0, \text{ i.e., no idle ambulances;} \\ -\tau_{W_a,\, Loc_{acc}(s)} & \text{if } Loc_{acc}(s) \neq 0 \ \wedge \ a \in A. \end{cases}$$

In our numerical experiments, we use $M = 15$ for the small region, and $M = 30$ for the region Flevoland. In both cases, $M > \tau_{i,j}, \forall i, j \in V$ holds. (In our implementation, time steps are equal to minutes.)

9.3.4 Transition Probabilities

Denote the probability of moving from state s to s', given that action a was chosen, as:

$$p^a(s,s'), \qquad \forall a \in \mathcal{A}_s, \qquad \forall s,s' \in S.$$

To compute the transition probabilities, note that the location of the next incident is independent of the set of idle ambulances. Thereto, $p^a(s,s')$ can be defined as a product of two probabilities. We write $p^a(s,s') = P_1(s') \cdot P_2^a(s,s')$, which stands for the probability that an incident happened at a specific location (P_1), and the probability that specific ambulances became available (P_2), respectively.

First of all, let us define $P_1(s')$. Since incidents occur according to a Poisson process, we can use the arrival rate λ (for an incident anywhere in the region) to obtain

$$P_1(s') = \begin{cases} \lambda \cdot d_{Loc_{acc}(s')} & \text{if } Loc_{acc}(s') \in V; \\ 1 - \lambda & \text{else.} \end{cases}$$

Note that the occurrence of incidents does not depend on the previous state (s).

Secondly, we need to model the process of ambulances that become busy or idle. For tractability, we will define our transition probabilities as if ambulances become idle according to a geometric distribution. In reality—and in our verification of the model—this is not the case, but since our objective is the long term average reward, this modeling choice should not have a negative impact [13]. Let us define a parameter $r \in [0,1]$, which represents the rate at which an ambulance becomes idle. We should set it in such a way, that the expected duration is equal to the average in practice. So this includes an average travel time, and an average time spent on scene. We add an average driving time to a hospital to that, as well as a realistic hospital drop off time—both multiplied with the probability that a patient needs to go to the hospital. For Dutch ambulances, this results in an average of roughly 38 min to become available after departing to an incident. For the geometric distribution, we know that the maximum likelihood estimate $\hat{r}$ is given by one divided by the sample mean. In this case, $\hat{r} = \frac{1}{38} \approx 0.0263$, which we use as the value for r in our numerical experiments.

We include a special definition if an ambulance was just dispatched. In such a case, the ambulance cannot be idle in the next time step. Furthermore, ambulances do not become busy, unless they have just been dispatched.

We now define

$$P_2^a(s,s') = \Pi_{i=1}^{|A|} P^a_{change}\big(idle_i(s), idle_i(s')\big), \quad \forall s,s' \in S,$$

where

$$P^a_{change}\big(idle_i(s), idle_i(s')\big) = \begin{cases} 1 & \text{if } a = i \wedge idle_i(s') = false; \\ 0 & \text{if } a = i \wedge idle_i(s') = true; \\ r & \text{if } a \neq i \wedge idle_i(s) = false \wedge idle_i(s') = true; \\ 1-r & \text{if } a \neq i \wedge idle_i(s) = false \wedge idle_i(s') = false; \\ 0 & \text{if } a \neq i \wedge idle_i(s) = true \wedge idle_i(s') = false; \\ 1 & \text{otherwise.} \end{cases} \tag{9.5}$$

9.3.5 Value Iteration

Now that we have defined the states, actions, rewards and transition probabilities, we can perform value iteration to solve the MDP. Value iteration, also known as backward induction, calculates a value $V(s)$ for each state $s \in S$. The optimal policy, i.e., the best action to take in each state, is the action that maximizes the expected value of the resulting state s'.

$V(s)$ is calculated iteratively, starting with an arbitrary value $V_0(s)\ \forall s \in S$. (In our case, we start with $V_0(s) = 0\ \forall s \in S$.) In each iteration i, one computes the values $V_i(s)$ given $V_{i-1}(s)\ \forall s \in S$ as follows.

$$V_i(s) := \max_{a \in \mathcal{A}_s} \{ \sum_{s'} p^a(s,s')(R(s,a) + V_{i-1}(s')) \} \tag{9.6}$$

This is known as the "Bellman equation" [3].

When the span of V_i, i.e., $\max V_i(s) - \min V_i(s)$, converges, the left-hand side becomes equal to the right-hand side in Eq. (9.6), except for an additive constant. After this convergence is reached, the value of $V(s)$ is equal to $V_i(s)\ \forall s \in S$.

We will extensively discuss the results of the MDP solution later in Sect. 9.6, but we now briefly state two observations. First, if an incident can not be reached in time anyway, the system is quite likely to choose an ambulance other than the closest idle one. The explanation is that, if the time threshold cannot be met, one might as well choose ambulance such that the remaining ambulances are in a favorable position with respect to possible future incidents. Second, whenever at least one vehicle is available within the target time, the MDP solution never prescribes to send a vehicle that is further away than the threshold.[1] That is to say, it does not appear to be beneficial to sacrifice performance now in the hope of achieving better performance later. We use these observations in the design of a dispatch heuristic that follows next.

[1] This holds for the realistic region that we implemented, but does not necessarily hold in general.

9.4 Solution Method: Dynamic MEXCLP Heuristic for Dispatching

In this section we describe a dispatch heuristic that was inspired by MDP solution above. The main benefit of the heuristic is that it can be computed in real time, for any number of vehicles and ambulance bases that is likely to occur in practice. Furthermore, the method is easy to implement. The heuristic is related to dynamic MEXCLP, also known as "DMEXCLP" [10], a heuristic that was originally designed to redeploy idle ambulances in real time.

The general idea is that, at any time, we can calculate the *coverage* provided by the currently idle ambulances. This results in a number that indicates how well we can serve the incidents that might occur in the (near) future.

More specifically, coverage is defined as in the MEXCLP model [7], that we will describe next.

9.4.1 Coverage According to the MEXCLP Model

In this section we briefly describe the objective of the well-known MEXCLP model. MEXCLP was originally designed to optimize the distribution of a limited number, say $|A|$, ambulances over a set of possible base locations W. Each ambulance is modeled to be unavailable with a pre-determined probability q, called the *busy fraction*. Consider a node $i \in V$ that is within range of k ambulances. The travel times $\tau_{i,j}$ $(i, j \in V)$ are assumed to be deterministic, which allow us to straightforwardly determine this number k. If we let d_i be the demand at node i, the expected covered demand of this vertex is $E_k = d_i(1 - q^k)$. Note that the marginal contribution of the kth ambulance to this expected value is $E_k - E_{k-1} = d_i(1 - q)q^{k-1}$. Furthermore, the model uses binary variables y_{ik} that are equal to 1 if and only if vertex $i \in V$ is within range of at least k ambulances. The objective of the MEXCLP model can now be written as:

$$\text{Maximize} \sum_{i \in V} \sum_{k=1}^{|A|} d_i(1 - q)q^{k-1} y_{ik}.$$

In [7], the author adds several constraints to ensure that the variables y_{ik} are set in a feasible manner. For our purpose, we do not need these constraints, as we shall determine how many ambulances are within reach of our demand points—the equivalent of y_{ik}—in a different way.

9.4.2 Applying MEXCLP to the Dispatch Process

The dispatch problem requires us to decide which (idle) ambulance to send, at the moment an incident occurs. Thereto, we compute the *marginal* coverage that

each ambulance provides for the region. The ambulance that provides the smallest marginal coverage, is the best choice for dispatch, in terms of remaining coverage for future incidents. However, this does not incorporate the desire to reach the current incident within target time T. We propose to combine the two objectives—reaching the incident in time and remaining a well-covered region—by always sending an ambulance that will reach the incident in time, if possible. This still leaves a certain amount of freedom in determining which *particular* ambulance to send.

The computations require information about the location of the (idle) ambulances. Denote this by $Loc(a)$ for all $a \in A_{idle}$. This model allows us to use the real positions of ambulances, which in practice may be determined by GPS signals. For simulation purposes, the current position of the ambulance while driving may be determined using, e.g., interpolation between the origin and destination, taking into account the travel speed. In either case, the location should be rounded to the nearest point in V, because travel times $\tau_{i,j}$ are only known between any $i, j \in V$.

Let A^+_{idle} denote the set of idle ambulances that are able to reach the incident in time. Similarly, let A^-_{idle} denote the set of idle ambulances that cannot reach the incident in time, which implies that $A^+_{idle} \cup A^-_{idle} = A_{idle}$. Then, if $A^+_{idle} \neq \emptyset$, we decide to dispatch a vehicle that will arrive within the threshold time, but chosen such that the coverage provided by the remaining idle vehicles is as large as possible:

$$\underset{x \in A^+_{idle}}{\arg\min} \sum_{i \in V} d_i (1-q) q^{k(i, A_{idle})-1} \cdot \mathbb{1}_{\tau_{Loc(x),i} \leq T}. \tag{9.7}$$

Otherwise, simply dispatch a vehicle such that the coverage provided by the remaining idle vehicles is as large as possible (without requiring an arrival within the threshold time):

$$\underset{x \in A^-_{idle}}{\arg\min} \sum_{i \in V} d_i (1-q) q^{k(i, A_{idle})-1} \cdot \mathbb{1}_{\tau_{Loc(x),i} \leq T}. \tag{9.8}$$

Note that in our notation, k is not an iterable, but a function of i and A_{idle}. $k(i, A_{idle})$ represents the number of idle ambulances that are currently within reach of vertex i. After choosing the locations of ambulances that one wishes to use—the real locations or the destinations—$k(i, A_{idle})$ can be counted in a straightforward manner.

9.5 Results: A Motivating Example

In this section, we consider a small region for which there is some intuition with respect to the best dispatch policy. We show that the intuitive dispatch policy that minimizes the fraction of late arrivals, is in fact obtained by both our solution methods (based on MDP and MEXCLP). We will address the alternative objective, i.e., minimizing the average response times, as well.

Figure 9.1 shows a toy example for demonstrative purposes. We let calls arrive according to a Poisson process with on average one incident per 45 min. Further-

more, incidents occur w.p. 0.1 in Town 1, and w.p. 0.9 in Town 2. Eighty percent of all incidents require transport to the hospital, which is located in Town 2.

9.5.1 Fraction of Late Arrivals

This section deals with minimizing the fraction of response times greater than 12 min. A quick analysis of the region in Fig. 9.1 leads to the observation that the "closest idle" dispatch strategy must be suboptimal. In order to serve as many incidents as possible within 12 min, it is evident that the optimal dispatch strategy should be as follows: when an incident occurs in Town 2, send ambulance 2 (if available). In all other cases, send ambulance 1 (if available). Both the MDP solution that attempts to minimize the fraction of late arrivals (with, e.g., $M = 15$), as well as the dispatch heuristic based on MEXCLP, lead to this policy (Fig. 9.2).

Note that in our model, it is mandatory to send an ambulance, if at least one is idle. Furthermore, we do not base our decision on the locations of idle ambulances (instead, we pretend they are at their destination, which is fixed for each ambulance). Therefore, in this example with 2 ambulances, one can describe a dispatch policy completely by defining which ambulance to send when both are idle, for each possible incident location. For an overview of the various policies, see Table 9.2.

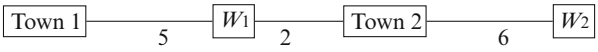


Fig. 9.1: A graph representation of the region. The numbers on the edges represent the driving times in minutes with siren turned on. W_1 and W_2 represent the base locations of ambulance 1 and 2, respectively. Incidents occur only in Town 1 and Town 2. The only hospital is located in Town 2

Solution method	$Loc_{acc} = Town1$	$Loc_{acc} = Town2$
MEXCLP(destination) heuristic	W_1	W_2
MDP(frac)	W_1	W_2
MDP(avg)	W_1	W_1

Table 9.2: An overview of the behavior of various dispatch policies when both ambulances are idle. The value in the table represents the base from which an ambulance should be dispatched

9.5.2 Average Response Time

We used the MDP method described in Sect. 9.3.3.2 to obtain a policy that should minimize the average response time, let us denote this policy by MDP(avg). We evaluate the performance of the obtained policy, again by simulating the EMS activities in the region. These simulations show that the MDP solution indeed reduces the average response time significantly, compared to the policy that minimizes the fraction of late arrivals (MDP(frac))—see Fig. 9.3.

9.6 Results: Region Flevoland

In this section, we simulate the redeployment method that we obtained from our MDP for a realistic problem instance. The Netherlands is divided in 24 regions, each operated by its own ambulance provider (see Fig. 9.4).

We modeled the region of Flevoland, which in practice is served by the ambulance provider "GGD Flevoland". For the parameters used in the implementation, see Table 9.3. This is a region with multiple hospitals, and for simplicity we assume that the patient is always transported to the nearest hospital, if necessary.

We estimated the arrival intensity for this region from historical data, and determined a reasonable number of vehicles that can serve this demand. Consequently,

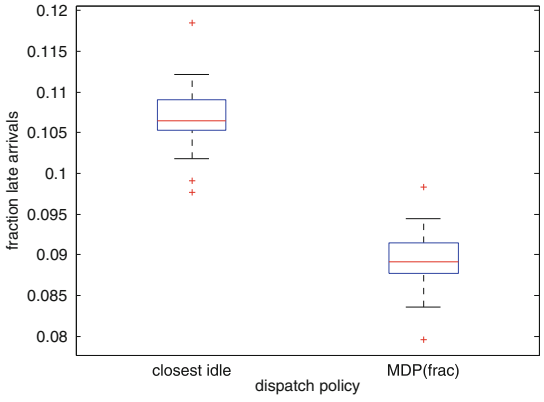


Fig. 9.2: Fraction of late arrivals as observed in a simulation of the small region. This figure shows the performance of the MDP solution that attempts to minimize the fraction of late arrivals (after value iteration converged). The performance is compared with the "closest idle" dispatch policy. Each policy was evaluated with 20 runs of 5000 simulated hours

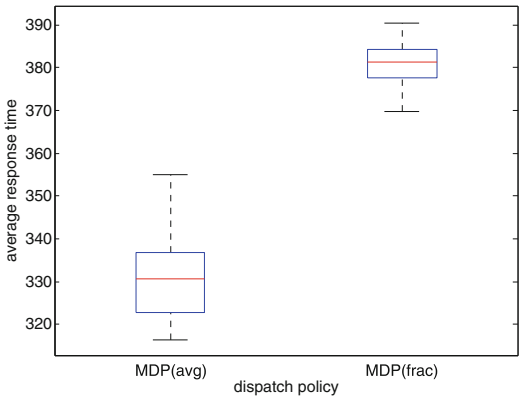


Fig. 9.3: The average response times in seconds, as observed in simulations of the small region. This figure shows the performance of the MDP solution that attempts to minimize the fraction of late arrivals versus the MDP solution that attempts to minimize the average response time (after value iteration has converged). Each policy was evaluated with 20 runs of 5000 simulated hours

we distributed the vehicles according to the solution of the (static) MEXCLP model, as described in Sect. 9.4.1. This model is generally assumed to give reasonably good solutions [16]. This static MEXCLP solution can be seen in Fig. 9.5.

Note that we used the fraction of inhabitants as our choice for d_i. In reality, the fraction of demand could differ from the fraction of inhabitants. However, the number of inhabitants is known with great accuracy, and this is a straightforward

Fig. 9.4: The 24 EMS regions in the Netherlands

Parameter	Magnitude	Choice
λ	1/29 min	A realistic rate for Flevoland.
A	8	A reasonable amount to serve the demand.
W_a for $a \in A$		Postal codes as depicted in Fig. 9.5.
V	91	All 4 digit postal codes in the region.
H	3	The hospitals within the region in 2013.
$\tau_{i,j}$		Driving times as estimated by the RIVM.
d_i		Fraction of inhabitants as known in 2009.

Table 9.3: Parameter choices for our implementation of the region of Flevoland

way to obtain a realistic setting. Furthermore, the analysis of robust optimization for uncertain ambulance demand in [11] indicates that we are likely to find good solutions, even if we make mistakes in our estimates for d_i.

In the Netherlands, the time target for the highest priority emergency calls is 15 min. Usually, 3 min are reserved for answering the call, therefore we choose to run our simulations with $T = 12$ min. The driving times for EMS vehicles between any two nodes in V were estimated by the Dutch National Institute for Public Health and the Environment (RIVM) in 2009. These are driving times with the siren turned on. For ambulance movements without siren, e.g., when repositioning, we used 0.9 times the speed with siren. The number of vehicles used in our implementation is such that value iteration is still tractable.

We simulate the problem as described in Sect. 9.2. In these simulations, ambulances that become idle are immediately dispatched to a waiting incident (if any),

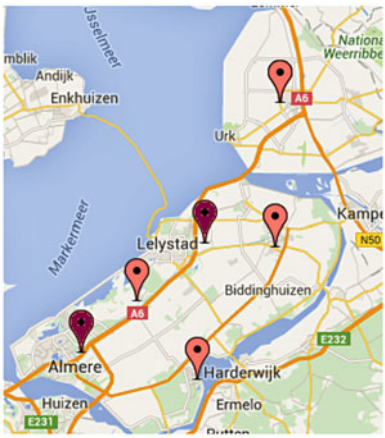


Fig. 9.5: Optimal distribution of 8 ambulances over the region Flevoland, according to the MEXCLP solution. Two vehicles are stationed at each of the *dark colored* bases, while 1 vehicle is present at the *lighter colored* locations

or head back to their home base. The locations we used as home bases are depicted in Fig. 9.5, and correspond to actual base locations in the EMS region. Ambulances that were dispatched while on the road, did not return to their base first.

In our simulation, $\tau_{onscene}$ is exponentially distributed with an expectation of 12 min. $\tau_{hospital}$ is drawn from a Weibull distribution with an expectation of approximately 15 min. More specifically, it has shape parameter 1.5 and scale parameter 18 (in minutes). We state these distributions for completeness, however, our numerical experiments indicate that the performance does not depend much on the chosen distribution for $\tau_{onscene}$ or $\tau_{hospital}$. In our simulations, patients need hospital treatment with probability 0.8. This value was estimated from Dutch data.

Note that $\tau_{onscene}$ or $\tau_{hospital}$ and the probability that a patient needs hospital treatment are not explicitly part of our solution methods. Instead, they subtly affect the busy fraction q (for the heuristic) or the transition probabilities with rate r (for the MDP).

9.6.1 Analysis of the MDP Solution for Flevoland

In this section, we highlight and explain some features of the MDP solution for the region Flevoland. In particular, we will focus on the states for which the MDP solution differs from the closest idle policy.

The output of the MDP is a table with the incident location, the status of the different ambulances (idle or not), and the optimal action. This output is a rather large table (with, in the case of Flevoland, 23,552 entries) that does not easily reveal insight into the optimal policy. To this end, we first reduced the size of the table: we filtered out all states for which no real decision has to be made (i.e., states in which no incident occurs, and states in which less than 2 ambulances are idle). This reduces the table size to 22,477. In 5583 of these states, the MDP solution is to send a vehicle *other* than the closest idle one (i.e., roughly 25%).

To understand in *which* states the MDP solution prescribes to send a vehicle other than the closest idle one, we used classification and regression trees (CART trees) on the table to find structure in the form of a decision tree. We used random forests to create the decision tree, since it is known that a basic CART has poor predictive performance. While bagging trees reduces the variance in the prediction, random forests also cancel any correlation structure in the generation of the trees that may be presenting while bagging.

The outcome that describes the MDP solution is a decision tree that divides the state space into three regions, see Fig. 9.6. For the red and the green region, whether or not the closest idle ambulance is sent, depends heavily on the availability of ambulances at base 1: if an incident occurred in the red region, one should *always* send an ambulance from base 1 if possible. For the red and green region combined, this same advice holds in 95% of the states.

If no ambulance at base 1 is idle, the MDP solution prescribes to deviate from the "closest idle" choice quite often, and especially so for the red and green regions: if there are more than two ambulances idle (but none at base 1), the dispatcher should deviate from the closest idle policy in 81% of the states. If exactly two ambulances are idle, this is still true for almost 50% of the states. This may be intuitively understood as follows. Since incidents on the red nodes can not be reached in time anyway, choosing an ambulance that is further away than the closest idle one results in a enlarged response time. However, using our objective of the fraction late arrivals, this is not a downside, since the incident could not be reached in time anyway. Therefore, an ambulance can be chosen such that the remaining ambulances are in a favorable position with respect to possible future incidents. Note that this is also the general idea that forms the basis of our MEXCLP dispatch heuristic.

For incidents on the blue nodes, the best decision according to the MDP is in roughly 70% of the states equal to the closest idle vehicle, and hardly depends on whether or not the ambulances at base 1 are idle.

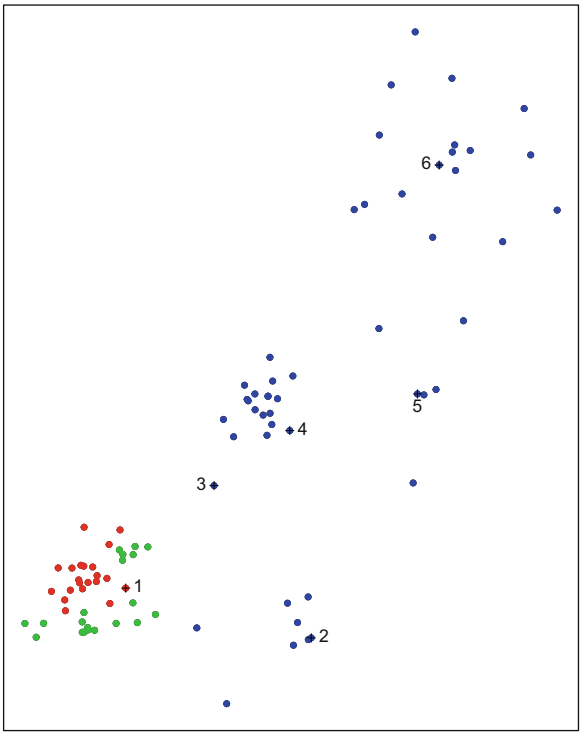


Fig. 9.6: Each node represents a postal code in Flevoland. Nodes with the same color have similar MDP solutions. The numbers indicate the bases. (Two vehicles are stationed at base numbers 1 and 4)

Out of all states for which at least one vehicle is available within the target time, the MDP solution *never* prescribes to send a vehicle that is further away than the threshold. That is to say, it does not appear to be beneficial to sacrifice performance now in the hope of achieving better performance later. This—again—is similar to the MEXCLP dispatch heuristic, because the heuristic also only sends a vehicle that will arrive late if there is no other option.

For this realistic region, value iteration took a long time to converge. Instead of waiting for convergence, one might also be interested in using the policy we get after a fixed number of value iterations. Figure 9.7 indicates that the performance after 3 iterations is already quite similar to the converged alternative.

9.6.2 Results

In this section, we show the results from our simulations of the EMS region of Flevoland.

We ran simulations using three different dispatch policies: the closest idle method, the MEXCLP-based heuristic and the MDP solution after convergence of the value iteration. Figure 9.8 compares their performance in terms of the observed fraction of response times larger than the threshold time.

The results show that the MDP solution that was designed to minimize the fraction of late arrivals has approximately the same performance[2] as the closest idle policy. Although this performance is perhaps somewhat worse than one may have hoped for, it is important to remember that the MDP has to decide which ambulance

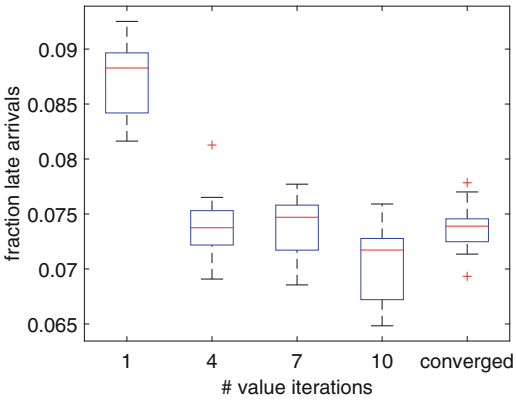


Fig. 9.7: The performance of the MDP solution for region Flevoland after 1, 4, 7 and 10 value iterations. Each policy was evaluated with 15 runs of 5000 simulated hours

[2] The fraction of late arrivals.

to send, based on which *base location* it belongs to. This, however, is not necessarily accurate since the ambulance may still be on the road (returning to base) at the moment of dispatch. In the simulations, this effect is accurately captured, but the MDP cannot (since keeping track of the true location of ambulances would lead to a state space explosion). Note that the "closest idle" method *does* have access to the real locations of vehicles at the time of dispatch, and thereby has a certain advantage.

As the MEXCLP-based dispatch heuristic has access to the real vehicle locations, it is not surprising that this method is able to perform better than the MDP solution. Consequently, the heuristic performs significantly better than the "closest idle" policy: it reduces the fraction of late arrivals from 7.22% to 6.26% on average: a relative improvement of 13%.

As mentioned earlier, the fraction of late arrivals is an important performance indicator for ambulance providers; however, one should also look at other aspects of the response time to make a well-informed decision on whether or not to implement a certain policy. We measure the average response time, as observed in our simulations: for the MDP solution, this is 483.9 s. The heuristic is slightly better with 478.9 s on average. The closest idle method outperforms both, resulting in an average response time of 409.3 s. Note that, with respect to this objective, the closest idle method is almost 15% better than our heuristic. In some sense, this is not surprising, but it does illustrate that our heuristic has such a strong focus on the fraction of late arrivals, that it becomes completely ignorant to the effect it has on the average response time (and the same holds for the MDP). This is an important observation that ambulance providers should keep in mind when they consider deviating from the closest idle policy.

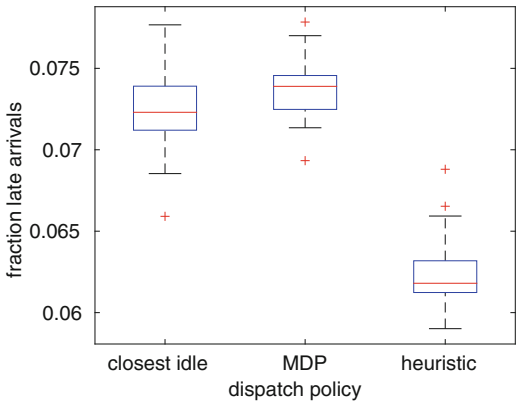


Fig. 9.8: Comparing the performance of the "closest idle" policy with the MDP solution and the Dynamic MEXCLP dispatch heuristic (where $q = 0.25$). Each policy was evaluated with 15 runs of 5000 simulated hours

9.7 Conclusion and Discussion

This chapter introduced two methods to obtain ambulance dispatch policies. Firstly, we modeled the ambulance dispatch problem as a Markov Decision Problem (MDP). This model is unique, in the sense that it is the first MDP in ambulance literature that keeps track of more than just the number of idle vehicles, without losing tractability for reasonably-sized ambulance fleets. Secondly, we introduced a heuristic that can easily be computed for any realistic size ambulance fleet.

Our result sheds new light on the popular belief that deviating from the closest idle dispatch policy cannot greatly improve the objective (the expected fraction of late arrivals). The above shown improvement of 13% was unexpectedly large. We consider this the main contribution of our work. Our methods yield in a great improvement in this KPI, however: one should be careful if one is also interested in other aspects of the response time. It is important to remember that our policies were designed with emphasis on the fraction of late arrivals only. Therefore, we do not claim that our dispatch policies are practically preferable over the closest idle policy, but we have shown that the argumentation for not using alternatives should be different. One should argue that we do not deviate from the closest idle policy, because we do not know how to do this while improving response times overall—and not because the alternatives fail to improve the fraction of late arrivals.

9.7.1 Further Research

One might consider making small changes to the MDP that could benefit the performance. For example, one idea is to artificially increase the rate with which busy ambulances become idle. This extra time would allow for ambulances to drive back to their home base, before the MDP considers them to be idle again. That way, we avoid the error where the MDP decides that an ambulance will reach an incident within the time threshold, but in fact the ambulance is still returning to base and happens to be further away from the incident. We suspect that this approach might give a small improvement; however, it should be noted that there is also a downside to making this change: ambulances are considered to be busy even though they are free, and hence suboptimal decisions will be made from time to time. In fact, sometimes an ambulance is *closer* than the MDP knows, because its previous patient was in the same area as the next patient.

Other changes could be, to add more information in the state about the ambulance's actual location while driving back to the home base. This, however, would lead to a state space explosion and the resulting model will—for realistically sized regions—most certainly not be solvable by value iteration.

Acknowledgements The authors of this chapter would like to thank the Dutch Public Ministry of Health (RIVM) for giving access to their estimated travel times for EMS vehicles in the Netherlands. This research was financed in part by Technology Foundation STW under contract 11,986, which we gratefully acknowledge.

Appendix: Notation

Notation in this chapter	Common notation
$\mathcal{A}_s$	$A(s)$
$R(s,a)$	$r^a(s)$
$p^a(s,s')$	$p(s\|s',a)$
$V_i(s)$	$V_t(s)$

References

1. R. Alanis, A. Ingolfsson, B. Kolfal, A Markov chain model for an EMS system with repositioning. Prod. Oper. Manag. **22**(1), 216–231 (2013)
2. R. Bellman, *Dynamic Programming* (Princeton University Press, Princeton, 1957)
3. R. Bellman, A Markovian decision process. J. Math. Mech. **6**(4), 679–684 (1957)
4. R. Bjarnason, P. Tadepalli, A. Fern, Simulation-based optimization of resource placement and emergency response, in *Proceedings of the Twenty-First Innovative Applications of Artificial Intelligence Conference*, 2009
5. G. Carter, J. Chaiken, E. Ignall, Response areas for two emergency units. Oper. Res. **20**(3), 571–594 (1972)
6. R.L. Church, C.S. Revelle, The maximal covering location problem. Pap. Reg. Sci. Assoc. **32**, 101–118 (1974)
7. M.S. Daskin, A maximum expected location model: formulation, properties and heuristic solution. Transp. Sci. **7**, 48–70 (1983)
8. S.F. Dean, Why the closest ambulance cannot be dispatched in an urban emergency medical services system. Prehosp. Disaster Med. **23**(02), 161–165 (2008)
9. J. Goldberg, R. Dietrich, J.M. Chen, M.G. Mitwasi, Validating and applying a model for locating emergency medical services in Tucson, AZ. Euro **34**, 308–324 (1990)
10. C.J. Jagtenberg, S. Bhulai, R.D. van der Mei, An efficient heuristic for real-time ambulance redeployment. Oper. Res. Health Care **4**, 27–35 (2015)
11. R.B.O. Kerkkamp, Optimising the deployment of emergency medical services. Master's thesis, Delft University of Technology, 2014
12. M.S. Maxwell, M. Restrepo, S.G. Henderson, H. Topaloglu, Approximate dynamic programming for ambulance redeployment. INFORMS J. Comput. **22**, 226–281 (2010)

13. M. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (Wiley, New York, 1994)
14. V. Schmid, Solving the dynamic ambulance relocation and dispatching problem using approximate dynamic programming. Eur. J. Oper. Res. **219**(3), 611–621 (2012)
15. C. Swoveland, D. Uyeno, I. Vertinsky, R. Vickson, Ambulance location: a probabilistic enumeration approach. Manag. Sci. **20**(4), 686–698 (1973)
16. P.L. van den Berg, J.T. van Essen, E.J. Harderwijk, Comparison of static ambulance location models. Under review, 2014
17. Y. Yue, L. Marla, R. Krishnan, An efficient simulation-based approach to ambulance fleet allocation and dynamic redeployment, in *AAAI Conference on Artificial Intelligence (AAAI)*, July 2012

Chapter 10
Blood Platelet Inventory Management

Rene Haijema, Nico M. van Dijk, and Jan van der Wal

Abstract This paper illustrates how MDP or Stochastic Dynamic Programming (SDP) can be used in practice for blood management at blood banks; both to set regular production quantities for perishable blood products (platelets) and how to do so in irregular periods (as holidays). The state space is too large to solve most practical problems using SDP. Nevertheless an SDP approach is still argued and shown to be most useful in combination with simulation. First the recipe for the stationary case is briefly reviewed as referred to earlier research. Here the regular production problem is periodic: demand and supply are weekday dependent but across weeks the problem is usually regarded as stationary. However, during a number of periods per year (roughly monthly) the problem is complicated by holiday periods and other events that imply non-stationary demand and production processes. This chapter particularly focuses on how to deal with the Blood Platelet (PPP) problem in non-stationary periods caused by holidays. How should production quantities anticipate holidays and how should production resume after holidays. The problem will therefore also be modelled as a finite horizon problem. To value products left in stock at the end of the horizon we propose to use the relative state values of the

R. Haijema (✉)
Operations Research and Logistics, Wageningen University, Wageningen, The Netherlands
e-mail: Rene.Haijema@wur.nl

N.M. van Dijk
Stochastic Operations Research, University of Twente, Enschede, The Netherlands
e-mail: n.m.vandijk@utwente.nl

J. van der Wal
Faculty of Economics and Business, University of Amsterdam, Amsterdam, The Netherlands

Stochastic Operations Research group, University of Twente, Enschede, The Netherlands

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_10

original periodic SDP. An optimal policy is derived by SDP. The structure of optimal policies is investigated by simulation. Next to its stationary results, as reported before, the combination of SDP and simulation so becomes of even more practical value to blood bank managers. Results show how outdating or product waste of blood platelets can be reduced from over 15% to 1% or even less, while maintaining shortage at a very low level.

Key words: Blood platelets, Perishable products, Blood inventory management, Non-stationary, Finite horizon

10.1 Introduction

10.1.1 Practical Motivation

An intriguing problem in blood inventory management is the production of platelet pools from voluntary whole blood donations. At the first production stage, most of the plasma and red blood cells (RBC) are extracted from a 500 ml whole blood donation. A mixture, called the buffy coat, of blood platelets and white cells is left. At a second stage the platelets of five different donors are extracted from the buffy coats and are pooled into a blood platelet pool (BPP), see Fig. 10.1.

In the Netherlands only 36% of buffy coats obtained after producing RBCs is used for the production of pools. So normally supply of buffy coats is not a problem. The demand for pools set by patients in hospitals remains to be highly uncertain, despite planned surgeries and transfusions. And clearly, as lives may be at risk, any risk for shortages should be avoided. However, there are three complications:

- As a first and major complication the quality of platelets pools (thrombocytes) deteriorate through time and have a limited shelf life of only 5 days (in some countries 7 days). Other blood products, such as RBC and plasma in contrast can be stored for months. As a consequence, blood platelets age and are discarded after passing the maximum shelf life. Spill may thus take place.

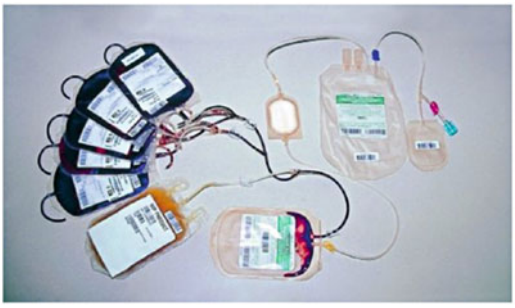


Fig. 10.1: One blood platelet pool (BPP) is made of platelets of 5 donors

- As a second complication also demand for blood platelets should be met according to compatibility rules for blood types (O, A, B, AB and the Rhesus-D factor).
- Finally, in practice there are short periods (breaks) such as around Christmas and Easter holidays during which there is no production.

Clearly, as the supply of blood takes place at voluntary basis, while high "costs" are involved in case of a shortage, blood platelets are to be considered as precious. Nevertheless, in practice (in both the US and Western Europe) a general figure for outdating is still about 15–20%. In what follows we refer to the problem of determining optimal production volumes as the platelet production problem (PPP).

10.1.2 SDP-Simulation Approach

In [6] and SDP-Simulation approaches is presented for the stationary PPP, which we will summarize in Sect. 10.3. In this chapter we show, in line with [7], how the non-stationary PPP can be solved using relative state values of stationary PPP.

The PPP is virtually stationary across weeks (with weekday dependent mean demand and production capacities), except for a number of periods within a year during which additional production breaks occur because of holidays, e.g. Easter, Christmas and New Year's Day. This kind of short production breaks, that occur roughly monthly, are known well in advance and should be anticipated adequately as they have substantial effect on inventory levels, and increase the risk of shortages.

- Setting the right production volumes to anticipate near-future production breaks is in general difficult. More formal support through an extended SDP-Simulation approach would thus be welcome to support blood bank managers.

In this paper, therefore, we focus on how to deal with the non-stationary periods. As argued in [7], we assume all demand to be met by issuing the oldest items in stock first, and the order quantity is not split into quantities per blood types. The maximal shelf life of BPPs is 5 days only. The method and results in this chapter are of interest to inventory managers at both blood banks and hospitals as well as to Operations Researchers.

10.1.3 Outline

In the next section we discuss some literature on the problem. In Sect. 10.3, we briefly summarize the steps involved in the SDP-Simulation approach that solves the stationary PPP. In Sect. 10.4 we present an extended SDP-Simulation approach to solve the PPP with periods where production and demand is non-stationary. In Sect. 10.5 we present case study results for a Dutch blood bank that faces short production breaks during Christmas, New Year's Day and Easter. Section 10.6 closes the chapter with conclusions and discussion. In the Appendix of this chapter follows a summary of notations

10.2 Literature

The production and inventory control of blood products have received considerable attention in both the traditional inventory operations research (OR) literature (e.g. [2, 3, 12–14]) and in the area of blood management (e.g. [8–11, 16]). In the OR literature, early dynamic programming (DP) formulations for blood inventory management already date back to the seventies (e.g. [3, 13]). Unfortunately, the platelet production problem is seriously hampered by its computational complexity as stated in [2]. All of these early studies assume a stationary demand distribution. Only in the last decade the problem is studied for periodic weekday dependent demand distributions, see [6]. The non-stationary problem is studied only in [7]. This chapter is largely based on that study and on Chap. 4 of [4].

For a general discussion of techniques for solving Markov decision processes for both stationary and non-stationary problems the reader is referred to textbooks like [15].

10.3 SDP-Simulation Approach for the Stationary PPP

10.3.1 Steps of SDP-Simulation Approach

In [6] a combined approach for the blood platelet inventory problem has therefore been followed, which combines OR and simulation by the following steps:

Step 1. Optimization model: First, a stochastic dynamic programming (SDP) formulation is provided, which neglects the existence of blood types. This latter assumptions will be validated in Step 5.

Step 2. Optimal solution: The dimension of the (SDP) formulation is then reduced (downsized) by aggregating the state space and demands so that the downsized (SDP) problem can be solved numerically (using successive approximation). That is, the optimal value and an optimal strategy is determined for the downsized SDP.

Step 3. Simulation for investigation: Then, as essential tying step, this optimal policy is (re)evaluated and run by simulation in order to investigate the structure of the optimal strategy. In this simulation one registers the frequency of (state, action)-pairs for the down-sized problem.

Step 4. Simulation for re-optimization: The results of step 3 are used to derive practical order rules, like improved base stock policies and to obtain nearly optimal parameter values. By a heuristic search procedure parameter values of these rules are fine tuned for the full-size problem.

Step 5. Simulation for validation: The quality (near-to-optimality) of this practical simple order-up-to strategy is evaluated by detailed simulation. In this step it is also justified, for Dutch blood banks, that blood types are ignored in the previous steps.

As the technical details of these steps are worked out in detail in [6], our focus in this chapter is on the SDP model (step 1) and on the extension of the SDP-simulation approach to solve the non-stationary problem.

10.3.2 Step 1: SDP Model for Stationary PPP

State

The state of the inventory model is the day of the week (d) and the number of products in stock of each age: $\mathbf{x} = (x_1, \ldots, x_{m-1})$, where m is the maximal shelf life of BPPs in days (which is usually 5 days).

Action

The sequential decision to set is the number of BPPs to produce (a). The action space $\mathcal{A}(d, \mathbf{x})$ is state dependent, as production capacity is not available during weekends.

State transition

The stock transition from weekday d to the next day, given that demand is met by a FIFO-issuing policy, follows from the stock transition function $y(\mathbf{x}, k, a)$ and depend on the initial stock $\mathbf{x}$, the demand k, and the production volume a. At the end of the day BPPs that have become outdated are disposed of. The transition probability relate to the demand distribution for weekday d is denoted by $p_d(k)$.

One-period costs

The direct cost to incur on a day depends on the state $(d, \mathbf{x})$ and the demand k. therefore we define $C(d, \mathbf{x}, k)$ is the costs to incur in on weekday d, when the demand on that day is k and the initial stock is $\mathbf{x}$. With the total stock level denoted by $x = \sum_{r=1}^{m} x_r$, the direct cost $C(d, \mathbf{x}, k)$ are:

$$C(d,\mathbf{x},k) = \begin{cases} c^{\mathrm{O}} \cdot (x_m - k)^+ & \text{outdating costs,} \\ +c^{\mathrm{S}} \cdot (k - \sum_{r=1}^{m} x_r)^+ & \text{shortage costs,} \\ +c^{\mathrm{H}} \cdot x & \text{holding costs.} \end{cases} \tag{10.1}$$

The expected direct costs to incur in state $(d,\mathbf{x})$ are simply:

$$\mathbb{E}C(d,\mathbf{x}) = \sum_k p_d(k) \cdot C(d,\mathbf{x},k). \tag{10.2}$$

Stochastic dynamic programming

Equation (10.3), presents the recursive form of the Bellman equation, which can be solved by a Successive Approximation algorithm, similar to SDP, unless the state space $\mathcal{X}$ is too large.

$$V_n(d,\mathbf{x}) = \min_{a \in \mathcal{A}(d,\mathbf{x})} \left(\mathbb{E}C(d,\mathbf{x}) + \sum_k p_d(k) \cdot V_{n-1}(d+1, y(\mathbf{x},k,a)) \right). \tag{10.3}$$

We start the SA algorithm by setting $V_0(1,\mathbf{x}) = 0$ for all states $\mathbf{x} \in \mathcal{X}$. The choice to start with $d = 1$ (i.e. a Monday) is arbitrary, we could as well choose any other weekday to start with. Next the state values $V_1(7,\mathbf{x})$, $V_2(6,\mathbf{x})$,…,$V_7(1,\mathbf{x})$,etc. are computed for all states $\mathbf{x}$.

The span$(\mathbf{V}_n - \mathbf{V}_{n-7})$ is checked every 7 iterations, hence at iteration $n = 7$, 14, 21, etc. Suppose $span(\mathbf{V}_n - \mathbf{V}_{n-7})$ is for the first time smaller than a pre-specified small value ε at iteration $N-7$, with N a multiple of 7 days. Optimal actions for each state $\mathbf{x}$ on Monday to Sunday are derived from Eq. (10.4) from the last 7 iterations ($n = N-6, N-5, \ldots, N$). Hence after N iterations an optimal stationary strategy is approximated by:

$$\pi(d,\mathbf{x}) = \arg \min_{a \in \mathcal{A}(d,\mathbf{x})} \sum_k p_d(k) V_{N-d}(d+1, y(\mathbf{x},k,a)). \tag{10.4}$$

In the last 7 iterations the optimal production strategy π is stored.

As N is a multiple of 7, the value vector $\mathbf{V}_N$ relates to a Monday. $\mathbf{V}_N$ is a relative value vector that can be used as terminal costs in solving the finite horizon problem of the non-stationary period.

Fig. 10.2: The Dutch blood banks divide The Netherlands in four regions

10.3.3 Case Studies

The approach has been validated and adopted by two of the four regional blood centers in The Netherlands, see Fig. 10.2.

Applying this combined SDP- simulation approach to data for two of four regional Dutch Blood Bank, the following conclusions are drawn, see [17]:

- A simple order-up-to rule could reduce the spill from roughly 15–20
- The combined SDP-Simulation approach led to an accuracy compared to the exact optimal value for the downsized problem within 1

In addition, a number of what-if analyses are performed such as with respect to:

- shelf life (4–7 days),
- issuing rules (FIFO/LIFO/FIFOR-r), and
- blood group compatibility/priorities.

In Sect. 10.4, we report detailed simulation results over a year that includes production breaks. In the next section we explain how the approach is extended.

10.4 Extended SDP-Simulation Approach for the Non-Stationary PPP

10.4.1 Problem: Non-Stationary Production Breaks

A practical question for production-inventory managers, and for blood bank managers in particular, that remained unanswered is: "How should one anticipate irregular production breaks like at Easter and Christmas?" When regular production stops for a few days, while demand continues (as usual) one should anticipate breaks by producing somewhat more some days before the break. Consequently the age distribution of the BPPs in stock is affected and thus the optimal production volumes immediately after a break might be different as well.

In the PPP most weeks are stochastically the same: the supply of whole blood (by voluntary donors) and the demand for BPPs are stationary processes. However, during a short holiday break production might be impossible, since donors do not show up.

As an example we consider, in Sect. 10.5, two cases of particular interest to blood bank managers:

1. A Christmas period (December 25 and 26) falling on Tuesday and Wednesday, followed by the New Year's Day (NYD) on the next Tuesday.
2. The 4-days Easter period from Good Friday to Easter Monday. In The Netherlands the (regular) production is stopped for four consecutive days, while the maximal shelf life of BPPs is only 5 days.

In practice it is difficult to find nearly optimal production volumes on days around holidays. Due to the occasional nature of these events, it is not possible getting experienced. Formal support is thus needed. In the next section we extend the SDP-Simulation approach to include non-stationary production breaks and apply it to the BPP production-inventory management.

10.4.2 Extended SDP-Simulation Approach

We illustrate the approach by including breaks at a Christmas period and an Easter weekend. As the time between these breaks is very long compared the maximal shelf life of the products (of 5 days only), one may analyze these periods independently of each other.

In Fig. 10.3 we illustrate our approach for the 4-days Easter weekend. The gray squared blocks on the time bar indicate production stops.

We model the problem as an infinite horizon problem, that consists of two parts. The first part is the irregular finite horizon period containing the non-stationary production break(s). The second part relates to the stationary infinite horizon problem.

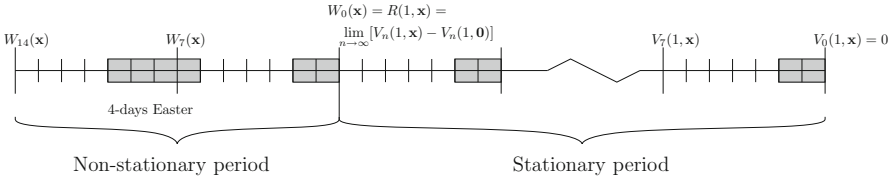


Fig. 10.3: The SDP-Simulation approach splits the horizon in two parts, since the system behaves stationary a few days after a break

In the example in Fig. 10.3, the age distribution of the BPPs in stock on the Tuesday immediately after the break differs from an ordinary Tuesday, the optimal ordering policy may be different than on ordinary Tuesdays. But we may assume that the system returns to the stationary problem a few days after the irregular period has elapsed. We assume that from the next Monday onwards the system behaves indeed stationary.

By stochastic dynamic programming (SDP) the optimal ordering strategy can be computed for both the stationary and the non-stationary problems. We first apply successive approximation to solve the stationary problem and thus obtain the optimal ordering strategy for regular weekdays, Monday to Friday. As a by-product, relative values of each stock state are available that can act as terminal costs to solve in a backward fashion the non-stationary problem. In the next two sections we formalize the approach.

10.4.3 Extension: Including Non-Stationary Periods

In a very similar way the ordering decisions for each day of the non-stationary finite horizon problem are computed. For example, in Fig. 10.3, the special period lasts 2 weeks (14 days) with the 4-days Easter weekend in the middle of the time interval. Note that we have chosen more or less arbitrarily that the finite horizon problem ends a few days after the break on a Monday morning.

For each of the 14 days of the special period the optimal production strategy is determined by Stochastic Dynamic Programming in a backward fashion, using the so-called relative values $R(1,\cdot)$ as terminal costs. The relative values $R(1,\mathbf{x})$ are used to compare stock states $\mathbf{x}$ within the same periodic class, namely the class related to Mondays, the value vector $V_N(1,\mathbf{x})$ can be used for this purpose. Instead of storing the optimal policy for each working day, we now store the policy for all 14 days of the non-stationary period.

The extended SDP-Simulation approach consists of the following five steps:

Step I. Compute relative values of the states for the stationary problem:

- First, the stationary problem is solved (maybe after scaling the problem), using Eqs. (10.3) and (10.4).
- Next, we choose an arbitrary reference state on Monday, say $(1,\mathbf{0})$, and compute for every possible stock state $\mathbf{x}$ on Monday the difference in expected future costs relative to this reference state:

$$R(1,\mathbf{x}) = \lim_{n\to\infty} [V_n(1,\mathbf{x}) - V_n(1,\mathbf{0})] \approx V_N(1,\mathbf{x}) - V_N(1,\mathbf{0}) \qquad (10.5)$$

These differences, $R(1,\cdot)$, are feasible relative values, when N is sufficiently large. $V_N(1,\mathbf{x})$ is finite when N is finite. Therefore setting $R(1,\mathbf{x}) = V_N(1,\mathbf{x})$ is also feasible.

Step II. Solving the non-stationary problem by SDP:
Let the irregular period last T days, with the last day being a Sunday. The days of this period are numbered backwards and denoted by t. $t = 1$ thus refers to the last day of the period (a Sunday) and day T is the first day of the finite horizon. Index t thus denotes the number of days to go until the end of the irregular period. In addition to the notation in Eqs. (10.3) and (10.4) we define:

- $p_t^{\text{irr}}(k)$ = the probability of a (composite) demand k on day t.
  If the demand remains stationary even during a break, then $p_t^{\text{irr}}(k)$ equals $p_d(k)$ for $d = 7-(t-1) \mod 7$.
- $\mathcal{A}'_t(\mathbf{x})$ = action space at day t as bounded by the (artificial) production and storage capacity. Clearly the action space depends on the stock state $\mathbf{x}$. If production is not possible on day t, $\mathcal{A}'_t(\mathbf{x}) = \{0\}$.
- $W_t(\mathbf{x})$ = the total expected costs under an optimal strategy from day t onwards when starting in inventory state $\mathbf{x}$ and at the end of the irregular period terminal cost are accounted.
- $W_0(\mathbf{x}) \equiv R(1,\mathbf{x})$.
- $\pi_t(\mathbf{x})$ = the optimal decision at day t given the inventory state is $\mathbf{x}$.

By stochastic dynamic programming one recursively computes and stores successively for $t = 1,2,\ldots,T$, for all states $\mathbf{x}$ in the state space $\mathcal{X}'(t)$:

$$W_t(\mathbf{x}) = \min_{a\in\mathcal{A}'_t(\mathbf{x})} \left(\mathbb{E}C_t(\mathbf{x}) + \sum_k p_d(k)\cdot W_{t-1}(y(\mathbf{x},k,a)) \right). \qquad (10.6)$$

with $\mathbb{E}C_t(\mathbf{x}) = \sum_k p_t^{\text{irr}}(k)\cdot C(d,\mathbf{x},k)$, in which d is the weekday related to t. The optimal ordering quantity on day t follows from

$$\pi_t(\mathbf{x}) = \arg\min_{a\in\mathcal{A}'_t(\mathbf{x})} \sum_k p_t^{\text{irr}}(k) W_{t-1}(y(\mathbf{x},k,a)). \qquad (10.7)$$

Step III. Read simple rule from simulation-based frequency table
Again the optimal strategy may be fairly complex. Hence simulation is used to investigate the structure of the optimal strategy for each day of the special period. That is: frequency tables for each day t of the irregular period are generated and, if applicable, an order-up-to S rule is read for each day of the week or any other appropriate ordering rule.

Step IV. Finally, by a detailed simulation program the rule is put to the test and its performance, in terms of outdating and shortage figures, is compared to the figures for the optimal stock-age-dependent strategy.

Step V. In case the initial problem is scaled the simple rule is simulated for the full-size problem after re-scaling the parameters of the simple rule.

10.5 Case Study: Optimal Policy Around Breaks

In this section we apply the extended SDP-Simulation approach using realistic data, as summarized in Sect. 10.5.1. The results for the problem around Christmas and New Year's Day are presented in Sect. 10.5.3. In Sect. 10.5.4 we discuss the 4-days Easter weekend. The results are integrated in Sect. 10.5.5. We will show that a simple rule applies even around breaks, although we do not provide a detailed sensitivity study as we have done in the previous chapter. We limit the illustration to the first four steps of the SDP-simulation approach.

10.5.1 Data

The case study is executed by using the data for one of the four Dutch Blood banks (Sanquin). The demand is met by issuing the oldest BPPs in stock (FIFO issuing). The mean demand for 144 BPPs a week is spread over Monday to Sunday as in Table 10.1. To make the SDP tractable we scale the problem by a factor 4, resulting in the mean demand figures in the last row of Table 10.1.

Table 10.1: Demand distributions: means and coefficients of variation (cv)

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Original mean	26	21	32	21	26	8	10
cv	0.28	0.31	0.25	0.31	0.28	0.50	0.45
Scaled mean	6.5	5.25	8	5.25	6.5	2	2.5

For each day of the week d we fit a discrete probability distribution $p_d(\cdot)$ on the mean demand and the reported coefficient of variation of the demand (cv). cv is set to 1.4 times the coefficient of variation when demand would have been Poisson distributed. Conform [1] the unscaled demand distributions are mixtures of two negative binomial distributions. For the scaled problem, the demand distributions are fitted using a mixture of two binomial distributions.

The other problem data for the unscaled case remain unchanged:

Annual demand	7488 BPPs or 1872 batches
Costs	outdating 150 per outdated BPP, shortage costs 750 per BPP short, all other costs are zero,
Production	Monday–Friday, but no during breaks
Maximal shelf life	$m = 5$ days.

For a high accuracy performances statistics will be obtained from 100 detailed simulation runs of 1000,000 weeks each.

10.5.2 Step I: Stationary Problem

After scaling the SDP and solving the scaled problem, we simulate the resulting optimal strategy. Table 10.2 presents the simulation-based frequency tables for Monday to Friday. In the first column we read the order-up-to levels related to the optimal actions. From the last column we read which level is most frequent. On Mondays the most-frequent order-up-to level is 21 and it fits to 59% of the one million states visited. Similarly, we find most-frequent order-up-to levels $(21, 21, 21, 19, 24)$ batches (of 4 BPPs) for Monday to Friday.

In Table 10.3 we compare the order-up-to rule with order-up-to levels fixed to $(21, 21, 21, 19, 24)$ against the optimal (age-dependent) strategy. The upper half of the table shows the characteristics of the optimal SDP strategy, the lower half shows the performance of a fixed replenishment rule. As expected from the results and the discussion in the previous chapter, the order-up-to S rule appears to perform nearly optimal. The absolute outdating and shortage figures per week should be related to a weekly demand of 36 batches of 4 pools. The annual results relate to an average annual demand for 1872 batches.

From the scaled SDP results we conclude that

- Compared to the current practice, it seems that the annual outdating can be reduced from about 15% to 0.2%, even when demand is more stochastic than Poisson.
- Shortages occur virtually never: only 0.04% of the total annual demand cannot be met immediately from stock.

Table 10.2: Simulation frequency tables of scaled SDP strategy with FIFO demand only

(a) (State, action)-frequency tables for 1,000,000 simulated Mondays

Stock x	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	Freq(S_1)
Up-to S_1																				
24													4	6	12	153	101	250	27	553
23												4	46	2906	17596	5508	1262			27322
22										5	1156	121403	79791	39802						242157
21								116497	152240	167708	154513									590958
20						39450	74336													113786
19					17209															17209
18				5976																5976
17			1657																	1657
16		332																		332
15	50																			50
14																				0
:																				:
0																				0
Freq(x)	50	332	1657	5976	17209	39450	74336	116497	152240	167713	155669	121407	79841	42714	17608	5661	1363	250	27	1000000

(b) (State, action)-frequency tables for 1,000,000 simulated Tuesdays

Stock x	0…7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	Freq(S_2)
Up-to S_2																		
26														1				1
25														1	1	3		5
24												2	6	42	33	22	1	106
23											5	277	622	898	247	16		2065
22									40	653	5055	15270	10982	2259	215			34474
21			786	9588	44321	83285	143525	191312	198489	159515	95150	32597	4316	427				963311
20		38																38
19																		0
:																		:
0																		0
Freq(x)	0	38	786	9588	44321	83285	143525	191312	198529	160168	100210	48146	15926	3628	496	41	1	1000000

(c) (State, action)-frequency tables for 1,000,000 simulated Wednesdays

Stock x	0…9	10	11	12	13	14	15	16	17	18	19	20	21	22	Freq(S_3)
Up-to S_3															
25													3		3
24									2		72	103	87	2	266
23								9	173	1314	4637	1865	592		8590
22						9	988	6708	49510	99883	31271	5334			193703
21				16565	59327	136626	214317	231213	136611						794659
20		167	2612												2779
19															0
:															:
0															0
Freq(x)	0	167	2612	16565	59327	136635	215305	237930	186296	101197	35980	7302	682	2	1000000

(d) (State, action)-frequency tables for 1,000,000 simulated Thursdays

Stock x	0…4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	Freq(S_4)
Up-to S_4																			
22																	3	6	9
21															106	753	204		1063
20											1156	7482	44639	30494	8368	931			93070
19					7114	24181	58816	110748	164500	196650	184235	129167	28890						904301
18			145	1401															1546
17		7																	7
16																			0
:																			:
1																			0
0																		4	4
Freq(x)	0	7	145	1401	7114	24181	58816	110748	164500	196650	185391	136649	73529	30494	8474	1684	207	10	1000000

(e) (State, action)-frequency tables for 1,000,000 simulated Fridays

Stock x	0…7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	Freq(S_5)
Up-to S_5																
28														1	2	3
27														2		2
26											30	262	223	70		585
25							1	378	2670	17755	20380	9471	1110			51765
24		141	2406	15686	57196	132318	210586	235857	185682	87884	19889					947645
23																0
:																:
0																0
Freq(x)	0	141	2406	15686	57196	132318	210587	236235	188352	105639	40299	9733	1333	73	2	1000000

Table 10.3: Order-up-to S rule vs SDP policy for a regular (stationary) week (statistics in batches of 4 pools obtained by simulation for 100 million weeks)

Weekday	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Weekly	Annual	
SDP policy										
# batches outdated	0	0.01	0.06	–[a]	–[a]	0	0.01	0.08	4.30	(0.23%)
# batches short	0.01	0	0	0	0	0	0	0.01	0.71	(0.04%)
Annual costs									4,698 euro	
Order-up-to S rule										
Levels S_d	21	21	21	19	24	–	–			
Goodness-of-fit	59%	96%	79%	90%	95%	–	–			
# batches outdated	0	0.01	0.06	–[a]	–[a]	0	0.01	0.08	4.18	(0.22%)
# batches short	0.01	0	0	0	0	0	0	0.01	0.74	(0.04%)
Annual costs									4,724 euro	

[a] Outdating must be zero on Thursday and Friday, as $m = 5$ and production stops in the weekends

- The order-up-to S policy as read from the frequency tables closely approximates the structure of the optimal strategy and is only 0.6% off from the optimal cost level.

For the unscaled case, the replenishment levels are re-scaled by multiplication with a factor 4.In the previous chapter we have already shown that the resulting order-up-to levels are nearly optimal. More results are found in [4–7, 17].

10.5.3 Steps II to IV: Christmas and New Year's Day

For example, we consider a year in which Christmas falls on Tuesday and Wednesday, and New Year's Day (NYD) on the next Tuesday. On these days (regular) production is stopped because of a lack of donors. As depicted on the timeline in Fig. 10.4 there is only a single day between the weekend and the two Christmas holidays. On this Monday one has to produce additional BPPs to anticipate the production stop for the next 2 days. In this section we consider a worst case where demand for platelet pools continues as usual.

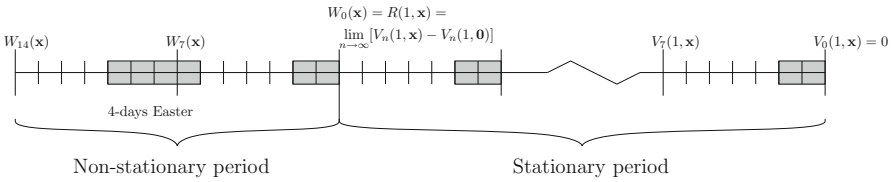


Fig. 10.4: The SDP-Simulation approach splits the horizon into two parts, as if the system behaves stationary a few days after a break

What to Expect?

When the production capacity on Monday is not restrictive, as in our case, then the production on Monday has to be set high enough to anticipate the production stop on the next 2 days. When the production capacity on Monday is normally high enough but too restrictive to anticipate the 2-day production stop, then the production volume on Friday is raised as well. When no holding costs and quality mismatch costs apply, the capacity restrictions will hardly affect the cost-level. Compared to the stationary case the optimal cost level does increase, since outdating and shortages are likely more prevalent. Shortages mostly occur on the Thursday after Christmas. Excessive production on the Monday before Christmas outdates the next Saturday.

We assume that the production and storage capacity are not restrictive. (Artificial bounds, which are set to make the state space finite, are set high enough to ensure that the optimal policy is not affected.) The length of the non-stationary period can then be reduced from 3 weeks (as in Fig. 10.4) to 2 weeks, as the problem in the first week maybe still stationary. When the production capacity on Wednesdays, January 2$^{\text{nd}}$, is not restrictive then the production strategy on Thursday January 3$^{\text{rd}}$, may differ only slightly from that on an ordinary Thursday. Moreover, given $m = 5$, no BPPs will expire on Thursday, as no BPPs in stock on Thursdays are older than 3 days.

Results for Optimal Strategy

Table 10.4 shows the impact of the irregular production breaks on the optimal strategy over a 10-days period from Monday (December, 24$^{\text{th}}$) to Wednesday (January, 2$^{\text{nd}}$). The results are presented in batches of 4 pools, since the SDP is scaled to reduce the state space. The average demand over the 10-days period is almost 56 batches (223 pools).

Table 10.4a shows that outdating for this 10-days period has increased from 0.15 batches (0.6 pools) to 0.77 batches (3.1 pools). Expected relative outdating thus is about 1.4%. Since the results relate to the optimal strategy one has to accept this increase, which is primarily due to the outdating of pools produced on the Monday before Christmas. Although not reported in the table, we have observed that only

0.55% of the BPPs (0.08 batch) produced on the Monday before News Year's Day becomes outdated on the next Saturday.

Table 10.4: Impact of production breaks around Christmas and New Year's Day

(a) Impact of breaks on outdating and shortages (in batches) under SDP policy

10-days period	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Mon	Tue	Wed	Total
Stationary case											
# batches outdated	0	0.01	0.06	–[a]	–[a]	0	0.01	0	0.01	0.06	0.15
# batches short	0.01	0	0	0	0	0	0	0.01	0	0	0.03
Irregular breaks		Dec-25	Dec-26						Jan-1		
# batches outdated	0	0.01	0.11	–[a]	–[a]	0.59	0	–[a]	–[a]	0.05	0.77
# batches short	0	0	0	0.06	0	0	0	0.01	0	0.01	0.08

[a] No outdating $m = 5$ days after weekends and holidays, since production is zero

(b) Impact of breaks on order-up-to levels S_t as read from the optimal SDP policy

10-days period	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Mon	Tue	Wed
Stationary case										
Order-up-to S_t	21	21	21	19	24	–	–	21	21	21
Goodness-of-fit	59%	96%	79%	90%	95%	–	–	59%	96%	79%
Irregular breaks		Dec-25	Dec-26						Jan-1	
Order-up-to S_t	31	–	–	20	24	–	–	27	–	21
Goodness-of-fit	98%	–	–	49%	44%	–	–	93%	–	48%

Order-up-to Rule vs Optimal Strategy

Since in practice one prefers a simple rule, we hope that an order-up-to S rule resembles the structure of the optimal policy. By simulation we generate (state, action)-frequency tables for each day of the special period in a similar way as for the stationary case. In Table 10.4b we report the most-frequent order-up-to levels for the 10-days period and compare them against the stationary ones. As expected the order-up-to levels on the Mondays before the two breaks are considerably higher than in the stationary case: e.g. 31 vs 21 before Christmas. Remarkably, an order-up-to S rule fits even better on Mondays just before a break, than on a regular Monday: 98% before Christmas versus 59% on the regular Mondays. On the days after a break the order-up-to S rule fits to almost 50% of the states visited, whereas in the stationary case this figure falls in 79–95%.

We evaluate the order-up-to S rule with the most-frequent order-up-to levels from the frequency tables by a long simulation (100 million replications) and compare its performance against the optimal strategy. (The results could be computed in an

exact way by solving the underlying Markov chains under modified cost structures.) Although the order-up-to S rule does not fit for 100% at each day, its performance is very close to optimal as reported in Table 10.5. Over the given 10-days period on average only 0.1 batch (out of the 56 batches demanded) cannot be met from stock: the shortage rate is thus less than 0.2%. Outdating is in the order of 1.3%.

Table 10.5: Order-up-to S vs SDP policy around Christmas and New Year's Day

10-days period	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Mon	Tue	Wed	Total
Irregular breaks		Dec-25	Dec-26						Jan-1		
SDP policy											
# batches outdated	0	0.01	0.11	–[a]	–[a]	0.59	–[a]	–[a]	0	0.05	0.77
# batches short	0	0	0	0.06	0	0	0	0.01	0	0.01	0.08
Order-up-to S rule											
Levels S_t	31	–	–	20	24	–	–	27	–	21	
# batches outdated	0	0.01	0.11	–[a]	–[a]	0.59	–[a]	–[a]	0.01	0.04	0.75
# batches short	0	0	0	0.06	0	0	0	0.02	0	0.01	0.10

[a] No outdating $m = 5$ days after weekends and holidays, since production is zero.

After re-scaling the order-up-to levels, one obtains a nearly optimal order-up-to S rule for the real-sized case. Given the robustness of the rule with respect to minor changes in the order-up-to levels as shown in the previous chapter we skip step V of our extended SDP-Simulation approach.

Conclusions: Results Breaks During Christmas and New Year's Day

It appears that:

- an order-up-to S policy with increased order-up-to levels for the Monday before Christmas and the Monday before New Year's Day is nearly optimal,
- age plays an more important role after Christmas and New Year's Day, but the absolute impact on outdating and shortages is relatively small, and
- outdating over a 10-days period, including the breaks, is as low as 1.4%, while shortages are less than 0.2%.

10.5.4 Steps II to IV: 4-Days Easter Weekend

The second example of a non-stationary problem with irregular production breaks is the 4-days Easter weekend, as depicted before in Fig. 10.3. The long weekend from Good Friday to Easter Monday is considerably more difficult, since the production is stopped for four consecutive days, while the shelf life is only 5 days and demand remains stationary. We assume that the production and storage capacity are not restrictive.

What to Expect?

Again, we expect that primarily the production volumes 1 day before the break, in this case on the Thursday before Good Friday, are increased dramatically to anticipate the production stops for the next 4 days. The available stock plus the production on Thursday should be enough to survive until the next Wednesday morning when new stock is released. The order-up-to level on Thursday before the break can be derived by the Newsboy model, since no BPPs will survive until Wednesday.

The marginal return of ordering z batches instead of $z-1$ batches is the savings on shortages. The marginal costs are an increase in the expected outdating costs. Let the stochastic variable Z denote the demand in batches over the 6-days period from Thursday to Tuesday, and $P(\cdot)$ is the cumulative distribution of Z. $P(\cdot)$ is obtained from the convolution of the six demand distributions. The optimal order-up-to level on Thursday is the greatest value of z for which the expected marginal savings on shortage costs is still larger than the expected marginal outdating costs, as reflected in Eq. (10.8).

$$c^{\mathrm{S}} \cdot P(Z \geq z) \geq c^{\mathrm{O}} \cdot P(Z \leq z-1) \tag{10.8}$$

Hence the best order-up-to level is the greatest z for which holds:

$$P(Z \leq z-1) \leq \frac{c^{\mathrm{S}}}{c^{\mathrm{S}} + c^{\mathrm{O}}}. \tag{10.9}$$

For the given demand distributions and the cost figures, the order-up-to level on the Thursday before the break is according to the Newsboy equation 31 batches. The production volume just before the break is thus likely much higher than on a regular Thursday. Consequently, outdating mostly happens 5 days later on Tuesday, just after the break. Since production stops from Friday to Monday, shortages primarily happen on Tuesday given that the production lead time is 1 day. We expect shortages and outdating to be far more prevalent than over the Christmas period.

On Tuesday morning after Easter Monday all products in stock, if any, are of the same age. The optimal production strategy is thus not stock-age-dependent. No BPPs produced before the break will survive until Wednesday morning, hence one may expect that the production volume on Tuesday is fixed to a target inventory level on Wednesday morning. Consequently, the optimal production volume on Wednesday is also fixed, as Tuesdays production becomes available only at the

start of Wednesday morning and all products in stock are of the same age. From Thursday onwards the optimal production strategy is again stock-age-dependent.

Results of Optimal Strategy

After scaling and solving the SDP, we can check our expectations. Through simulation we generate frequency tables from which we read the structure of the optimal strategy. A selection of them is found in Table 10.6. As expected and argued before, according to the upward diagonal in Table 10.6a, a fixed production volume applies on the Tuesday just after Easter Monday. All batches produced before the break will not survive until the Wednesday morning after Easter.

Since all stock present on Tuesday morning after Easter Monday will perish the same day, the initial stock on Wednesday morning consists only of the 15 batches produced on Tuesday. From Table 10.6a we observe that a fixed order-up-to level of 21 batches applies on Wednesday, which implies a fixed production volume of 6 batches. Note that an order-up-to level of 21 batches corresponds to the stationary order-up-to level reported in Table 10.4b.

The order-up-to level on Thursday after the break equals 19, which corresponds to the stationary order-up-to level. This illustrates that the stationary order-up-to levels apply from Wednesday onwards. Although not reported the outdating and shortages figures from the Thursday after the break onwards do not differ significantly from those on regular days. One positive exception, not visible in the table, is that outdating on the Sunday after the break is significantly lower than usual.

The optimal production policy on Thursday prior to Good Friday resembles for virtually 100% an order-up-to S rule with fixed order-up-to level 32. This is in-line with our expectations: the newsboy model suggest an order-up-to level of 31 batches. As the initial stock may not survive till the next ordering moment, the best order-up-to level is 1 higher. The order-up-to level 32 is 13 batches higher than on a regular Thursday (see Sect. 10.5.2).

Just as for the Christmas and New Year's Day period, on the day before a production break an order-up-to S rule (with increased levels) fits even better than in the stationary case. Apparently the age-distribution of the stock is less relevant on a day prior to a production break.

Order-up-to Rule vs Optimal Strategy

The structure of the optimal policy seems thus, again, to be very well presented by an order-up-to S rule. In Table 10.7 we report over a 10-days period, including the Easter weekend, the outdating and shortage volumes around Easter under both the

Table 10.6: Frequency tables from 1 million simulations of SDP policy around Easter

(a) (State, action)-frequency table for a Tuesday after Easter Monday

Stock x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	Freq(S)
Up-to S																							
36																						1	1
35																					20		20
34																				176			176
33																			933				933
32																		3578					3578
31																	10375						10375
30																23749							23749
29															43775								43775
28														68301									68301
27													91544										91544
26												110320											110320
25											116201												116201
24										118217													118217
23									110287														110287
22								94938															94938
21							74669																74669
20						53775																	53775
19					35596																		35596
18				21466																			21466
17			11893																				11893
16		5758																					5758
15	4428																						4428
14																							0
⋮																							⋮
0																							0
Freq(s)	4428	5758	11893	21466	35596	53775	74669	94938	110287	118217	116201	110320	91544	68301	43775	23749	10375	3578	933	176	20	1	1000000

(b) (State, action)-frequency table for a Wednesday after Easter Monday

Stock x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	Freq(S)
Up-to S																							
21																1000000							1000000
20																							0
⋮																							⋮
0																							0
Freq(s)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1000000

(c) (State, action)-frequency table for a Thursday after Easter Monday

Stock x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	Freq(S)
Up-to S																							
21																					245		245
20																		27771	8545	1830			38146
19																	66652						959504
18							260	1830								122189							2090
17									8547	27770	66651	122191	174559	196384	174561								0
⋮																							⋮
1																							0
0																						15	15
Freq(s)	0	0	0	0	0	0	260	1830	8547	27770	66651	122191	174559	196384	174561	122189	66652	27771	8545	1830	245	15	1000000

optimal production policies and the order-up-to S rule. In the table Good Friday and Easter Monday are abbreviated by GF respectively EM.

In the last column of Table 10.7 we observe that the order-up-to S rule performs almost equally well as the optimal SDP policy. As expected, outdating and shortages happen mostly on the Tuesday after Easter Monday. Under both strategies on average approximately 4.3 batches will outdate and stock falls on average 0.28 batches short. The average demand over the 10-days period is on average almost 56 batches and the average production to cover the demand over this period is roughly 60 batches. Relative outdating over the 10-days period is thus $\frac{4.3}{60} = 7.2\%$. The shortage rate over the 10-days period is $\frac{0.28}{56} = 0.5\%$.

Conclusions: Results 4-Day Easter Weekend

The major conclusions over a 10-days period that includes the 4-days Easter weekend are:

- A simple replenishment rule performs nearly optimal and results in:
  - an outdating figure of only 7.2%,
  - shortages to be less than 1%.
- Outdating and shortages happen mostly on the Tuesday after Easter Monday, indicating the difficulty in anticipating a 4-days production stop when the maximal shelf life is only 5 days.

Table 10.7: Order-up-to S rule vs SDP policy around Good Friday and Easter Monday

10-days period	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Mon	Tue	Wed	Total
Irregular breaks					GF			EM			
SDP policy											
# batches outdated	0	0.01	0.06	–[a]	–[a]	0	0.03	0.07	4.16	–[a]	4.34
# batches short	0.01	0	0	0	0	0	0	0	0.26	0	0.28
Order-up-to S rule											
Levels S_t	23	22	24	32	–	–	–	–	15[b]	21	
Goodness-of-fit	44%	75%	64%	100%	–	–	–	–	100%[b]	100%	
# batches outdated	0	0.01	0.06	–[a]	–[a]	0	0.03	0.04	4.19	–[a]	4.33
# batches short	0.01	0	0	0	0	0	0	0	0.26	0	0.28

[a] No outdating $5(=m)$ days after weekends and holidays, since production is zero
[b] A fixed production volume applies since all batches in stock are outdated at the end of the day

- The production levels on the Thursday before and the Tuesday after the weekend are considerably higher than in the stationary case.
  - On the Tuesday after Easter Monday a fixed production quantity applies, since pools produced before Good Friday will not survive until Wednesday.
  - Shortly after the break (from Wednesday onwards) the stationary order-up-to *S* rule resumes as a very good approximation of the optimal SDP policy.

10.5.5 Conclusions: Extended SDP-Simulation Approach

Extended SDP-Simulation Approach

The SDP-Simulation approach can be applied to solve the order problem of perishables with a (short) fixed shelf life. We have extended the approach such that it can deal with (non-stationary) production breaks. It appears to be a powerful approach for deriving an optimal stock-age-dependent (scaled) policy and for investigating the structure such that a practical rule can be derived from it.

Results Over a Year Including Easter and Christmas

We have tested the approach on a PPP using realistic data for a single category of demand. For the PPP under consideration even around breaks simple order-up-to *S* rules apply and optimal order-up-to levels are easily read from simulation-based frequency tables.

By combining the results from the previous sections, the following conclusions can be drawn concerning the performance of the SDP-Simulation approach over a year which includes the breaks during Christmas, New Year's Day and the 4-day Easter weekend:

- Average annual shortage = 1.1 batch = 4.2 pools $<0.1\%$
- Average annual outdating = 9.0 batches = 36 pools $<1\%$

Compared to the current practice the potential savings are substantial: it seems that the current outdating figure of 15–20% can be reduced to less than 1%, while shortages arise only a few times per year.

10.6 Discussion and Conclusions

In this chapter, an SDP-Simulation approach is extended such that it can also deal with non-stationary periods (e.g. additional production breaks during holidays), in a further stationary horizon. In particular, we have investigated the impact on the

ordering strategy of some production breaks that occur during a year: i.e. on Good Friday, Easter Monday, the two Christmas days and New Years Day.

The combination of Simulation with SDP allow to solve real size problems. Other issuance policy than FIFO can be included. The complexity would increase when including blood types, however, as argued in [5], for the Dutch case with ample supply of whole blood, the distinction of blood types can usually be neglected.

Based on data from the Dutch blood banks (Sanguin), we draw the following conclusions:

- Additional outdating and shortages on the day(s) after a production break are to be accepted even under the truly-optimal SDP policy.
- An order-up-to S rule resembles the optimal policy even better on the day prior to an additional production break than on ordinary weekdays.
- When the production break last $m-1$ days, the order problem on the day before the break is a single period problems. The best order-up-to level S just before the break is then a bit higher than the one suggested by a Newsboy equation, as the initial stock will not survive until the next order moment.
- Simulation results over a year, including the production breaks during Christmas, New Year's Day and the 4-days Easter weekend, indicate that compared to the current practice overall outdating and shortage figures can still be reduced significantly:
  - outdating from 15–20% to less than 1%,
  - shortages from about 1% to less than 0.1%).

Acknowledgements The authors thank Cees Smit-Sibinga and Wim de Kort for supporting this research and its implementation at the Dutch blood banks of Sanquin.

Appendix: Notation

This section describes the relation of the terminology in this chapter related to the general notation in the book.

$\mathbf{s}$	$=(d,\mathbf{x})=$ is the state on weekday d and stock state $\mathbf{x}$ with elements
x_r	= number of products in stock with remaining shelf life of r days
$\pi(\mathbf{s})$	optimal policy = number of products to order on week day d if stock state is $\mathbf{x}$
$c(\mathbf{s})$	$\mathbb{E}C(d,\mathbf{x})$ single period (expected) cost in state $\mathbf{s}$
$V_n(\mathbf{s})$	value function storing (relative) expected costs of an optimal policy over n days

References

1. I. Adan, M. van Eenige, J. Resing, Fitting discrete distributions on the first two moments. Prob. Eng. Info. Sci. **9**(04), 623–632 (1995)
2. J.T. Blake, S. Thompson, S. Smith, D. Anderson, R. Arellano, D. Bernard, Using dynamic programming to optimize the platelet supply chain in nova scotia, in ed. By M. Dlouhý, *Proceedings of the 29th Meeting of the European Working Group on Operational Research Applied to Health Services, Prague, Czech Republic* (2003), pp. 47–65
3. M.A. Cohen, W.P. Pierskalla, Perishable inventory theory and its application to blood bank management. Technical report, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, Illinois, 1974
4. R. Haijema, Solving Large Structured markov decision problems for perishable inventory management and traffic control. Ph.D. thesis, University of Amsterdam - Tinbergen Institute - Amsterdam School of Economics, 2008
5. R. Haijema, J. van der Wal, N.M. van Dijk, Blood platelet production: a high-dimensional perishable inventory problem, in *Operations Research Proceedings 2004*, ed. By H. Fleuren, D. den Hertog, P. Kort (Springer, Berlin, 2005), pp. 84–92
6. R. Haijema, J. van der Wal, N.M. van Dijk, Blood platelet production: optimization by dynamic programming and simulation. Comput. Oper. Res. **34**(3), 760–779 (2007). doi:10.1016/j.cor.2005.03.023
7. R. Haijema, N.M. van Dijk, J. van der Wal, C. Smit Sibinga, Blood platelet production with breaks: optimization by SDP and simulation. Int. J. Prod. Econ. **121**, 467–473 (2009). doi:10.1016/j.ijpe.2006.11026
8. S.M. Hesse, C. Coullard, M.S. Daskin, A.P. Hurter, A case study in platelet inventory management, in *Proceedings of the Sixth Annual Industrial Engineering Research Conference, Miami Beach, Florida*, 1997
9. A.J. Katz, C.W. Carter, P. Saxton, J. Blutt, R.M. Kakaya, Simulation analysis of platelets production and inventory management. Vox Sang. **44**, 31–36 (1983)
10. R.E. Ledman, N. Groh, Platelet production planning to ensure availability while minimizing outdating. Transfusion **24**(6), 532–533 (1984)
11. J. McCullough, J. Undis, J.W. Allen Jr., Platelet production and inventory management, *Platelet Physiology and Transfusion*, in ed. By D.M. Mallory (American Association of Blood Banks, Washington, DC, 1978), pp. 17–38
12. S. Nahmias, Perishable inventory theory: a review. Oper. Res. **30**, 680–708 (1982)
13. W.P. Pierskalla, C.D. Roach, Optimal issuing policies for perishable inventory. Manag. Sci. **18**, 603–614 (1972)
14. G.P. Prastacos, Blood inventory management: an overview of theory and practice. Manag. Sci. **30**(7), 777–800 (1984)
15. M.L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (Wiley, New York, 2014)

16. V. Sirelson, E. Brodheim, A computer planning model for blood platelet production and distribution. Comput. Methods Prog. Biomed. **35**, 279–291 (1991)
17. N.M. van Dijk, R. Haijema, J. van der Wal, C. Smit Sibinga, Blood platelet production: a formal approach for practical optimization. Transfusion **49**(3), 411–420 (2009). doi: 10.1111/j.1537-2995.2008.01996.x

Part III
Transportation

Chapter 11
Stochastic Dynamic Programming for Noise Load Management

T.R. Meerburg, Richard J. Boucherie, and M.J.A.L. van Kraaij

Abstract Noise load reduction is among the primary performance targets for some airports. For airports with a complex lay-out of runways, runway selection may then be carried out via a preference list, an ordered set of runway combinations such that the higher on the list a runway combination, the better this combination is for reducing noise load. The highest safe runway combination in the list will actually be used. The optimal preference list selection minimises the probability of exceeding the noise load limit at the end of the aviation year. This paper formulates the preference list selection problem in the framework of Stochastic Dynamic Programming that enables determining an optimal strategy for the monthly preference list selection problem taking into account future and unpredictable weather conditions, as well as safety and efficiency restrictions. The resulting SDP has a finite horizon (aviation year), continuous state space (accumulated noise load), time-inhomogeneous transition densities (monthly weather conditions) and one-step rewards zero. For numerical evaluation of the optimal strategy, we have discretised the state space. In addition, to reduce the size of the state space we have lumped into a single state those states that lie outside a cone of states that may achieve the noise load restrictions. Our results indicate that the SDP approach allows for optimal preference list selection taking into account uncertain weather conditions.

Key words: Noise load management, Stochastic dynamic programming, Airport, Runway preference list selection

T.R. Meerburg • M.J.A.L. van Kraaij
Air Traffic Control the Netherlands, Schiphol, the Netherlands
e-mail: t.r.meerburg@lvnl.nl; m.vankraaij@lvnl.nl

R.J. Boucherie (✉)
Stochastic Operations Research, University of Twente, Enschede, The Netherlands
e-mail: r.j.boucherie@utwente.nl

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_11

11.1 Introduction

Safety and *efficiency* are two main aspects that determine capacity and characterise the service provided by air traffic control centers worldwide. Protection of the *environment*—including pollution, smell, third party risk and especially noise—increasingly influences airport operations. Airports where noise load is taken into account include Amsterdam Airport Schiphol, Logan International Airport in Boston and John F. Kennedy International Airport in New York. In the Netherlands, the Dutch Aviation Act restricts runway and route usage, limits the total noise load produced by aircraft during one aviation year, and enforces a certain distribution of noise load over enforcement points in the direct surroundings of Amsterdam Airport Schiphol [8]. If the cumulative noise load in an aviation year exceeds its maximum, the civil aviation authority may impose severe sanctions, such as fines, temporary closure of a runway or a reduction in the number of aircraft movements. Hence, noise load needs to be managed.

For airports with a complex lay-out of runways not all runway combinations are feasible due to safety and efficiency restrictions, e.g., due to weather criteria. When more than one runway combination is available, at Schiphol the one is utilised that is most preferred with respect to noise load management [1]. This paper proposes an efficient noise load management scheme that is based on *preference lists:* an ordered list of runway combinations that can be used under different weather conditions. Each preference list has its main contribution to the noise load in a particular part of the vicinity of the airport due to the ordering of runway combinations. Upon selection of a preference list, in daily operation the safe runway combination that is highest on the list is used. Each month, based on the realised noise load, flight schedules, and available runway combinations, an optimal preference list is generated that is expected to produce the most balanced accumulation of noise load with the aim to avoid exceeding noise load limits while taking into account uncertain future weather conditions.

This paper formulates the preference list selection problem in the framework of Stochastic Dynamic Programming [10]. The resulting SDP has a finite horizon (the aviation year), continuous state space (the accumulated noise load) and time-inhomogeneous transition densities (the monthly weather conditions). The optimal strategy minimises the probability of exceeding the noise load limit at the end of the aviation year. To cast the preference list selection problem into the SDP framework, for each month and each preference list we have evaluated the distribution of the accumulated noise load given the weather conditions. As we are interested in the probabilities of exceeding the noise load limit, the one step rewards are zero. To enforce the Markov structure, we have assumed that the accumulated noise loads in subsequent months are independent random variables. For numerical evaluation of the optimal strategy for preference list selection, we have discretised the state space. In addition, to reduce the size of the state space we have lumped into a single state those states that lie outside a cone of states that may achieve the noise load restrictions. In our case study for Schiphol, we compare the optimal preference list obtained via our SDP approach with current heuristic approaches and we investigate the influence of the number of decision epochs.

The paper is organised as follows. Section 11.2 presents the noise load management problem. Our SDP framework is described in Sect. 11.3. Implementation issues due to e.g. the size of the problem are discussed in Sect. 11.4. Section 11.5 presents a feasibility study. Finally, Sect. 11.6 provides conclusions and recommendations.

11.2 Noise Load Management at Amsterdam Airport Schiphol

Noise load is a main steering parameter for the selection of preferred runway combinations at Amsterdam Airport Schiphol. To monitor the realised noise load a number of enforcement points is placed in the vicinity of Schiphol with locations chosen to represent an appropriate noise contour surrounding the airport relevant for the population in its vicinity, see Fig. 11.1. For each aircraft, the noise load contribution to these enforcement points—expressed in A-weighted decibels dB(A), which is an expression for the relative loudness of sounds in air as perceived by the human ear—is calculated based on its flight path, and added to the already realised noise load [13]. In principle, excess in a single enforcement point in an aviation year (November 1st–October 31st) results in government measures irrespective of the amount of dB(A) by which the noise limit is exceeded. We have considered eight preference lists that are available for implementation at Schiphol consisting of lists of runway combinations per period of the day. Different preference lists distribute noise load differently, so that selection of preference lists is used as a steering measure to balance the noise load over the enforcement points, or since weather conditions are uncertain, to minimise the risk of exceeding the noise load limit in one or more enforcement points.

Schiphol has five major runways, that allow for different runway combinations. The supply of inbound and outbound traffic varies over time, resulting in several inbound peaks, outbound peaks, intermediate-periods and a night-period. For these different periods, different runway combinations are utilised. During peak periods three runways are in use. During an inbound peak, arriving traffic is handled on two runways and departing traffic on one. During an outbound peak, arriving traffic is handled on one runway and departing traffic on two. During off-peak and night periods one runway is in use for arriving traffic and one for departing traffic. A complete description of the preference lists is omitted. For details, see [9]. The primary runways for the highest preference position on the lists and the enforcement points that are expected to be affected most by the implemented preference list are shown in Table 11.1. For a general idea, preference lists 1–4 contribute relatively more in the northern enforcement points, preference lists 5–8 more in the southern points, since the inbound runway combination with the highest preference contribute in those designated directions. Details of preference list 1 are provided in the Appendix.

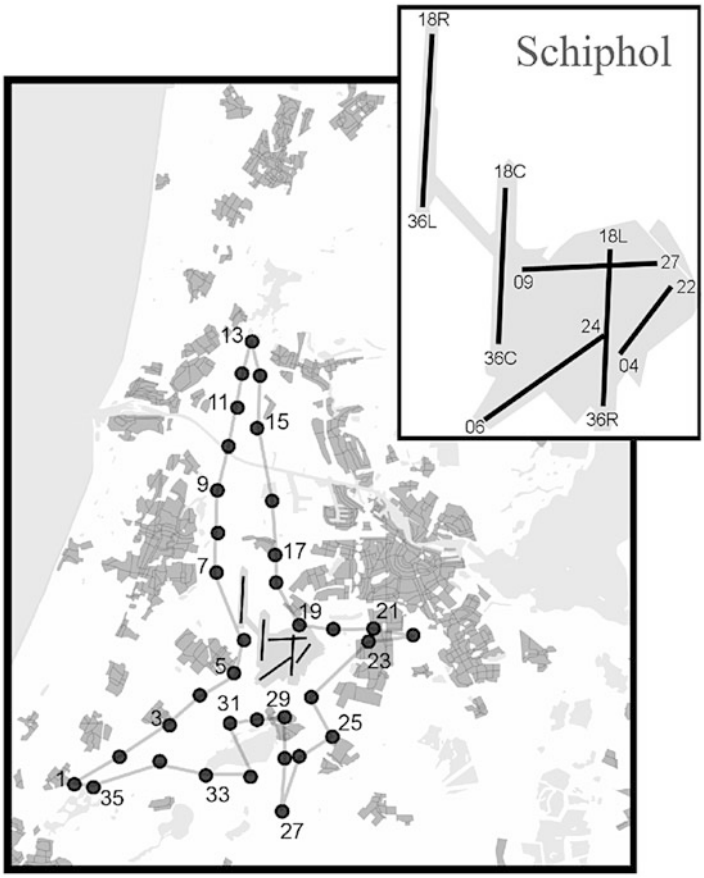


Fig. 11.1: 35 enforcement points and runway layout of Schiphol

Pref.	Inbound peak			Outbound peak			Largest exp. contrib.				
list	Dep.	Arrival		Departure		Arr.	enforcement points				
1	36L	06	36R	36L	36C	06	18	19	8	9	21
2	36L	06	36R	36L	09	06	21	20	19	22	9
3	36L	06	36R	24	36L	27	22	21	9	8	7
4	36L	06	36R	36L	09	06	21	20	19	22	23
5	24	18R	18C	24	18L	18R	5	4	19	25	31
6	24	18R	18C	24	18L	18R	5	4	19	31	22
7	24	18R	18C	24	18L	18R	21	20	19	22	25
8	24	18R	18C	24	18L	18R	21	20	19	5	4

Table 11.1: Available preference lists (primary runways)

The control strategy used until 2006 uses the average monthly weather conditions, obtained from data collected since 1971. Taking into account the current noise load realisation, a preference list is determined. This heuristic and its decision-making process of a stylised problem is discussed in [4]. It takes into account the expected noise load, but does not take into account (i) a possible change in future weather conditions, i.e., the probability distribution of the weather development, and (ii) the possible adaptation of the preference list due to these weather developments. In short, the heuristic obtains a single optimal preference list for each subsequent month in the remaining decision period based on constant (average) weather conditions, and the current noise load realisation. In this paper, we recast the preference list selection problem into the framework of Stochastic Dynamic Programming that each month generates an optimal preference list that takes into account (i) the probabilistic nature of the weather conditions, and (ii) the resulting possibly different preference lists used at subsequent decision epochs.

11.3 SDP for Noise Load Optimisation

Preference lists are selected each month based on the realised noise load at the enforcement points, the available runway combinations, and the weather conditions in the remaining part of the aviation year. In an SDP setting, the realised noise load at the beginning of month n (the decision epoch) is the *state* of the system at time n. A *decision* at time n is the selection of a preference list from all possible preference lists, i.e., from all available ordered sets of available runway combinations. For each decision, the evolution of the noise load in month n is determined by the weather conditions. As these are uncertain, this evolution is characterised by transition probabilities. The goal is to select—in accordance with the monthly regulation decisions—each month, in each state, a preference list such that the probability of exceeding the noise load limit at one or more enforcement points at the end of month 12 (the end of the aviation year) is minimised.

In a slightly more general setting we have the following mathematical characterisation of our decision process, that also allows for investigation of additional decision epochs and an arbitrary set of enforcement points. Let $\mathbb{R}_+ = [0,\infty)$, the non-negative real numbers, K the number of enforcement points, and N the number of decision epochs. Further, let X_n denote the random variable recording the noise load realisations at the enforcement points in period or stage n (between decision epoch n and $n+1$), and let

$\mathbf{i} = (i_1, \ldots, i_K) \in \mathbb{R}_+^K$: the state with a noise load realisation i_k in enforcement point k, $k = 1, \ldots, K$,
$S \subseteq \mathbb{R}_+^K$, $S_n \subseteq \mathbb{R}_+^K$: the set of noise load realisations, and those at decision epoch n
D_n : the set of preference lists available at decision epoch n, $n = 1, \ldots, N$, that may vary over the months due to e.g. planned maintenance,
$P_{n,d}(\mathbf{x}) = P(X_n \leq \mathbf{x}|d)$: the probability distribution of the noise load contribution in stage n when preference list d is selected at stage n,
$p_{n,d}(\mathbf{x})$: its density,
$f_n(\mathbf{i})$: the minimal probability of exceeding the noise load limit at the end of period N when the noise load realisation at stage n is i.

Note that $P_{n,d}(\cdot)$ does not depend on the noise load realisation, but only on the month and selected preference list. This is due to the weather conditions that vary over the months of the year. The function $f_n(\mathbf{i})$ satisfies the following recursion

$$f_n(\mathbf{i}) = \min_{d \in D_n} \left[\int f_{n+1}(\mathbf{x}+\mathbf{i}) p_{n,d}(\mathbf{x}) \mathrm{d}\mathbf{x} \right], \quad \mathbf{i} \in S_n, \tag{11.1}$$

where $\mathbf{x}+\mathbf{i}$ is component wise addition of $\mathbf{x}$ and $\mathbf{i}$. To see this, note that $f_{n+1}(\mathbf{x}+\mathbf{i})$ is the minimal probability of exceeding the noise load limit at the end of period N when the noise load realisation at stage $n+1$ is $\mathbf{x}+\mathbf{i}$ (that is when the optimal decision is selected starting at stage $n+1$), and that, for given $d \in D_n$, $p_{n,d}(\mathbf{x})$ is the density of the noise load contribution in period n, and therefore the integral $\int f_{n+1}(\mathbf{x}+\mathbf{i})$ $p_{n,d}(\mathbf{x})\mathrm{d}\mathbf{x}$ is the probability of exceeding the noise load limit at the end of period N following decision d taken at stage n. As a consequence, the minimal probability of exceeding the noise load limit starting at stage n is obtained by selecting the optimal decision $d \in D_n$. The *optimal control strategy* $\pi = (\pi_1, \ldots, \pi_N)$ is a set of decision rules $\pi_n : S_n \to D_n$ that assigns an optimal decision $d \in D_n$ to each state $\mathbf{i} \in S_n$ in stage n, $n = 1, \ldots, N$.

The recursion (11.1) is clearly a backward recursion: given $f_{n+1}(\mathbf{i})$ is known, the optimal decision that minimises the integral can be determined. Thus, the recursion requires the starting values for $f_{N+1}(\mathbf{i})$:

$$f_{N+1}(\mathbf{i}) = \begin{cases} 0 \text{ if } \mathbf{i} \leq L_{max} \\ 1 \text{ if } \mathbf{i} \not\leq L_{max} \end{cases} \qquad \mathbf{i} \in S_{N+1}, \tag{11.2}$$

where the inequalities are component wise, that is $\mathbf{i} \leq L_{max}$ if and only if $i_k \leq L_{max,k}$ for all enforcement points k, and $\mathbf{i} \not\leq L_{max}$ when there is some k for which the noise load restriction is violated. The minimum probability of exceeding the noise load limit in a year is $f_1(0)$, and once $f_1(0)$ is determined, also the optimal control strategy is determined.

Our SDP has the following characteristics:

- it has finite horizon as it models a single aviation year,

- the state space is continuous since the states represent the accumulated noise load measured in dB(A),
- the transition densities are time-inhomogeneous since the weather conditions depend on the specific months,
- the one-step rewards are zero since our optimal strategy minimises the probability of exceeding the noise load limit at the end of the aviation year.

Determining the optimal strategy requires the transition probabilities $P_{n,d}(\mathbf{x})$. The formulation of our optimisation problem involves the assumption that the contributions of the noise load in subsequent months are independent random variables. This is clearly an assumption, since the weather conditions today are perhaps the best ingredients for a forecast for the weather tomorrow. However, on a monthly scale these effects of dependence are marginal. Thus, we have identified all ingredients for determining the optimal control strategy $\pi = (\pi_1, \ldots, \pi_N)$.

Some numerical issues remain. For example, determining the empirical distribution $P_{n,d}(\mathbf{x})$ or its density $p_{n,d}(\mathbf{x})$ is far from obvious. Below, we will introduce a discretised approach. This discretised approach also induces a discretised version of the optimisation problem (11.1).

11.4 Numerical Approach

Multi-dimensional continuous-state dynamic programming problems are a huge challenge, in spite of the growth in computing power. The number of enforcement points and decision epochs prohibits an exact solution within a reasonable amount of computing time. Therefore, we propose a discrete approximation. This approximation involves discretisation of the state space, that in turn also calls for a discretisation of the transition probabilities.

11.4.1 Transition Probabilities

The monthly noise load contribution consists of a large number of small contributions by different aircraft. These noise load contributions depend on the weather conditions, which are highly unpredictable. The distribution further depends on the preference list d and on the month (since supply of traffic and weather conditions differ between seasons) represented by stage n. This brings us clearly in a setting that allows us to invoke the Central Limit Theorem, implying that under preference list d the monthly noise load contribution X_n has a multivariate normal distribution with K variates (enforcement points), $\mathcal{N}(\mu_{n,d}, \Sigma_{n,d})$, with $\mu_{n,d} \in \mathbb{R}^K$ the expected values of the K variates (the expected noise load contribution at the enforcement points), and $\Sigma_{n,d} \in \mathbb{R}^{K \times K}$ their covariance matrix.

11.4.2 Discretisation

To facilitate numerical evaluation of the risks of exceeding the noise load limits, we have discretised the state space S by forming a grid with distance ε among grid points in each dimension, i.e., we have divided the noise load in intervals of width ε. Selecting ε we have to balance between sufficient accuracy (small ε) and computational efficiency (large ε). To this end, in our numerical experiments below we have selected a discretisation step of 2% (noise load relative to its limit), that is, for enforcement point k the noise load interval $(0, L_{max,k})$ is divided into 50 intervals of equal width. The appeal of this discretisation is that it reduces the continuous problem to a problem with a finite number of states that can be solved numerically and approximates the solution of the SDP [6].

We will take the grid points to be the center of the interval. A grid point $\hat{\mathbf{i}} = (\hat{i}_1, \ldots, \hat{i}_K) \in \mathbb{N}_+^K$ corresponds to a noise load realisation in the hypercube $(\hat{i}_k\varepsilon - \varepsilon/2, \hat{i}_k\varepsilon + \varepsilon/2), k = 1, \ldots, K$. The state space $\hat{S} = \{\hat{\mathbf{i}} \cdot \varepsilon : \hat{\mathbf{i}} \in \mathbb{N}_+^K\}$ is the discretised set of noise load realisations. All states with noise load realisation exceeding the limit at some enforcement point may be lumped into a single state, since the probability of exceeding is 1 in such states irrespective of the number of enforcement points exceeding the limit, and the amount of overshoot.

Discrete transition probabilities are obtained by integrating the transition density $p_{n,d}(\mathbf{x})$ over the discretisation increment:

$$\hat{P}_{n,d}(\hat{\mathbf{i}}) = \int_{\hat{\mathbf{i}}-\frac{\varepsilon}{2}}^{\hat{\mathbf{i}}+\frac{\varepsilon}{2}} p_{n,d}(\mathbf{x})\mathrm{d}\mathbf{x}. \tag{11.3}$$

Notice that this is a K-dimensional integral, that can numerically readily be evaluated via Monte-Carlo summation, see e.g. [5].

Let $\hat{f}_n(\hat{\mathbf{i}})$ be the minimal probability of exceeding the noise load limit at the end of period N when the noise load realisation at stage n is $\hat{\mathbf{i}}$ in the discretised setting. Clearly, $\hat{f}_n(\hat{\mathbf{i}})$ satisfies the discretised equivalent of (11.1):

$$\hat{f}_n(\hat{\mathbf{i}}) = \min_{d \in D_n} \left[\sum_{\{\mathbf{x} | \mathbf{x}+\hat{\mathbf{i}} \in \hat{S}\}} \hat{f}_{n+1}(\mathbf{x}+\hat{\mathbf{i}}) \cdot \hat{P}_{n,d}(\mathbf{x}) \right], \qquad \hat{\mathbf{i}} \in \hat{S}. \tag{11.4}$$

The recursion requires starting values $\hat{f}_{N+1}(\hat{\mathbf{i}})$ by analogy with those of the continuous state problem.

11.5 Numerical Results

This section presents a feasibility study of our optimization approach with actual traffic and weather data that includes a comparison to the original heuristic, and an investigation into the possible benefit of an increased number of decision epochs.

The examples used in this section are for illustration purposes only. Consequently, results presented in this section cannot be used for other purposes like commercialization and decision-making.

We present a series of experiments that closely resemble the actual behaviour of the system for Schiphol. The input parameters are estimated from data generated in DAISY [3], an airport environment toolkit developed by Frontier Information Technologies BV that produces values for noise load in enforcement points given a volume of traffic, a preference list, a period, and weather conditions. Simulations for all combinations of N stages with $|D|$ preference lists were performed using the since 1971 recorded meteorological data. From these simulations we have obtained an estimate for the mean $\mu_{n,d}$ and covariance matrix $\Sigma_{n,d}$ for all combinations of stage n and preference list d for a year of uninterrupted operation (hence, no runway closure due to maintenance or other restrictions), with runway and route configurations and traffic supply scenario equivalent to that of 2006 (436,731 flight movements).

Research leading to this paper has been intended as a feasibility study for an improved preference list selection process. As a consequence, an extensive and optimised numerical program has not been developed. We have implemented our algorithm in the numerical computing environment Matlab [12]. The running time of our algorithm is exponential in the number of enforcement points. Therefore, we have restricted our analysis to sub-models containing three or four enforcement points. Our program is implemented on a 1.7 GHz PC, resulting in running times of a couple of days for four enforcement points with a discretisation interval of 2%. Given these limitations, our results indicate that our approach is indeed able to obtain preference lists with low probabilities of exceeding noise load limits.

11.5.1 Probability of Exceeding the Noise Load Limit

This section studies the optimal control strategy based on a representative set of three enforcement points. Selecting subsets of enforcement points may introduce undesired behaviour in the other enforcement points, e.g., the optimal preference list may completely avoid these three points distributing noise load over the remaining points. Therefore, we have selected eight different representative sets of enforcement points consisting of enforcement points that turn out to be most sensitive for noise load excess and covering different directions relative to Schiphol. Furthermore, we have restricted the available preference lists to the eight lists of Table 11.1 capturing the main contribution to the modeled enforcement points.

Table 11.2 provides for each set of enforcement points the probability of exceeding the noise load restriction, as well as the preference list in month 1 (November). Note that a complete description of the optimal strategy π is rather involved, since it requires for each realised noise load and each month a specification of the preference list.

Noise load management with $N = 12$		
Enforcement points	Probability of exceeding	Preference list month 1
5 19 21	0.1010	1
5 21 25	0.0368	5
9 19 25	0.0151	3
9 22 31	0.0847	5
18 19 21	0.1626	1
18 19 25	0.0267	3
18 19 31	0.0234	3
19 21 31	0.1519	1

Table 11.2: Results for optimal strategy

As a sanity check, for each optimal strategy (that is the strategy corresponding to the set of enforcement points), we have also considered the noise load contribution at the remaining 32 enforcement points when implementing the optimal strategy obtained for three enforcement points. This has shown only a slight excess in some non-modeled enforcement points, which leads us to believe that our subsets were well-chosen in accordance with the actual behavior of our system. Moreover, our algorithm seems to capture the behavior of the noise load problem and yields a good policy.

11.5.2 Comparison with the Heuristic

This section provides a comparison between our algorithm and the current heuristic in a typical situation occurring towards the end of an aviation year when a small number of enforcement points is at risk of exceeding the noise load limits, while the other points are far from reaching that limit. We will first provide some insight in the difference between the strategies, then compare our strategy with the heuristic described in [4], and finally investigate some possible trade-offs in sub-optimal preference list selection to allow for a selection of preference lists that better matches the requests from e.g. air lines.

The optimal decision π' of the heuristic that takes into account the expected noise load only can also be obtained from our algorithm when we reduce the state spaces S to only contain the mean noise load realisation. Our algorithm takes into account all possible future noise load realisations. Now consider a scenario with $N = 4$ decision epochs, and a set of noise load limits in three modeled enforcement points 9, 22 and 31 that all have consumed 68% of their noise load limits at the time of month 8, leaving 32% to be allocated in the remaining 4 months, and the 'certainty' that all other enforcement points will not exceed. Table 11.3 below gives the optimal control strategy π (note that $\pi_1 = 5$), and its probability of exceeding the noise load

limit. The control strategy for the current heuristic is $\pi' = \pi^{(1)} = (5,1,5,1)$. Other heuristic strategies $\pi^{(2)}$–$\pi^{(6)}$ using the preference lists 1 and 5 from $\pi^{(1)}$ are also evaluated to test for optimal selection of the ordering of preference lists from the heuristic π'. Clearly, π substantially outperforms all these heuristics.

Noise load management with $N = 4$					
Control strategy					Probability of exceeding
π	5	π_2	π_3	π_4	0.1113
$\pi^{(1)}$	5	1	5	1	0.2705
$\pi^{(2)}$	5	5	1	1	0.2747
$\pi^{(3)}$	5	1	1	5	0.2412
$\pi^{(4)}$	1	1	5	5	0.2433
$\pi^{(5)}$	1	5	1	5	0.2428
$\pi^{(6)}$	1	5	5	1	0.2773

Table 11.3: Results for optimal strategy and current heuristic

In addition to obtaining an optimal control strategy and corresponding probability of exceeding the noise load restriction, our algorithm also allows for fast evaluation of proposed changes to the optimal strategy. Deliberate sub-optimal decisions can be made to satisfy interests and demands of other aviation parties. Table 11.4 provides results when preference list 2 is forced for a number of months and the optimal strategy is used for the remaining months. As can be observed from the table, using preference list 2 for 1 month results in a doubling the probability of exceeding the noise load limits. Our algorithm allows for a trade-off of the results of deviating from the optimal strategy.

11.5.3 Increasing the Number of Decision Epochs

It may be beneficial to increase the frequency of preference list updates when this considerably affects the probability of exceeding the noise load limits. A trade-off has to be made between the overhead of preference list modification, and the reduction in the probability of exceeding the noise load limits. This trade-off is beyond the scope of this paper. As we see from Table 11.5, the probability of exceeding the noise load limits is significantly lowered by doubling the number of decision epochs, recall Table 11.2.

Noise load management with $N = 4$					
Control strategy					Probability of exceeding
π	π_1	π_2	π_3	π_4	0.1113
$\pi^{(7)}$	2	$\pi_2^{(7)}$	$\pi_3^{(7)}$	$\pi_4^{(7)}$	0.2037
$\pi^{(8)}$	2	2	$\pi_3^{(8)}$	$\pi_4^{(8)}$	0.4518
$\pi^{(9)}$	2	2	2	$\pi_4^{(9)}$	0.7340
$\pi^{(10)}$	2	2	2	2	0.9744
$\pi^{(11)}$	$\pi_1^{(11)}$	2	2	2	0.8391
$\pi^{(12)}$	$\pi_1^{(12)}$	$\pi_2^{(12)}$	2	2	0.4856
$\pi^{(13)}$	$\pi_1^{(13)}$	$\pi_2^{(13)}$	$\pi_3^{(13)}$	2	0.2567

Table 11.4: Effect of forcing a decision for a number of months

Noise load management with $N = 24$			
enforcement points			Probability of exceeding	Improvement w.r.t. $N = 12$ (%)
5	19	21	0.0397	61
5	21	25	0.0249	32
9	19	25	0.0042	72
9	22	31	0.0293	65
18	19	21	0.0141	13
18	19	25	0.0221	17
18	19	31	0.0198	15
19	21	31	0.1298	15

Table 11.5: Effect of doubling the number of decision epochs

11.6 Discussion

The optimal selection of preference lists is of utmost importance for efficient allocation of Schiphol's capacity considering noise load restrictions. To facilitate a decision between dynamic preference list selection and a fixed preference list, the benefit of optimal preference list selection must be quantified. To this end, this paper has described a monthly preference list selection process, that takes into account the probabilistic nature of the weather and resulting possibly different preference lists used at subsequent decision epochs. Results from our feasibility study indicate that our algorithm yields an adequate preference list selection strategy and allows for discrimination among different control strategies. As such, the effect of decisions deviating from the optimal strategy has been investigated. The operational feasibility of a monthly change to the preference list was not part of this study.

Stochastic Dynamic Programming adequately captures the structure of the decision problem as it allows for decisions under uncertainty for a finite horizon (yearly) problem. Although SDP seems to be the adequate approach, it also has a major drawback in that it is highly sensitive to the state space explosion problem making SDP too large to handle numerically. To alleviate this problem, concepts from the theory of huge Markov chains may be invoked to improve efficiency. Alternatively, other approximation methods and heuristics, such as neuro-dynamic programming [2], reinforcement learning [11] or discrete event simulation [7] may be used. A detailed study of the applicability of such methods is beyond the scope of this feasibility study that has shown that our Stochastic Dynamic Programming based algorithm allows for optimal preference list selection taking into account uncertain weather conditions.

Preference lists are used at several airports with a more complex lay-out of runways. Airports with layout of similar complexity as Schiphol's layout, such as Logan International Airport in Boston and John F. Kennedy International Airport in New York, also make use of a preference list to control noise load in its surroundings. Since noise abatement is becoming increasingly important in the aviation sector, more airports may develop similar noise load management procedures to control noise load in their environments. Due to the versatility of the proposed model, it can easily be implemented for any airport.

Appendix

Description of Preference Lists

During peak periods three runways are in use. During an inbound peak, arriving traffic is handled on two runways and departing traffic on one. During an outbound peak, arriving traffic is handled on one runway and departing traffic on two. During off-peak and night periods one runway is in use for arriving traffic and one for departing traffic.

For a general idea, preference lists 1 to 4 contribute relatively more in the northern enforcement points, preference lists 5 to 8 more in the southern points, since the inbound runway combination with the highest preference contribute in those designated directions. Noise load contributions of these variations are similar, but may avoid excessive contribution in a critical enforcement point. As an illustration, preference list 1 is provided below. Note that the first row of this list is included in Table 11.1. For details, see [9].

Preference list 1 contributes relatively more in the northern enforcement points.

Preference list 1										
	Inbound peak			Outbound peak			Off peak		Night period	
Nr.	Dep.	Arr.		Dep.		Arr.	Dep.	Arr.	Dep.	Arr.
1	36L	06	36R	36L	36C	06	36L	06	36L	06
2	24	18R	18C	24	18L	18R	24	18R	24	18R
3	18L	18R	18C	18L	18C	18R	18C	18R	18C	18R
4	36L	36R	36C	36L	36C	36R	36L	36C	36L	36C
5	24	27	18R	36L	09	06	09	18R	06	06
6	24	18R	22	24	36L	27	09	06	24	18C
7	18L	18R	22	24	18L	27	24	27	24	27
8	09	06	09	24	27	27	36L	27	36L	27
9	36L	06		36L		06	24	24	24	24
10	24	18R		24		18R	27	27	09	09
11	18L	18R		18L		18R	09	09	06	06
12	36L	36R		36L		36R	24	22	09	18R
13	09	18R		09		18R	06	06		
14	09	06		09		06				
15	24	27		24		27				
16	36L	27		36L		27				
17	24	22		24		22				
18	27	27		27		27				
19	09	09		09		09				
20	24	24		24		24				
21	06	06		06		06				

Notation

$\mathbf{i} = (i_1, \ldots, i_K) \in \mathbb{R}_+^K$	The state with a noise load realisation i_k in enforcement point k, $k = 1, \ldots, K$,
$S \subseteq \mathbb{R}_+^K$, $S_n \subseteq \mathbb{R}_+^K$	The set of noise load realisations, and those at decision epoch n
D_n	The set of preference lists available at decision epoch n, $n = 1, \ldots, N$, that may vary over the months due to E.g. planned maintenance,
$P_{n,d}(\mathbf{x}) = P(X_n \leq \mathbf{x} \mid d)$	The probability distribution of the noise load contribution in stage n when preference list d is selected at stage n,
$p_{n,d}(\mathbf{x})$	Its density,
$f_n(\mathbf{i})$	The minimal probability of exceeding the noise load limit at the end of period N when the noise load realisation at stage n is i

References

1. Aeronautical Information Publication, the Netherlands, http://www.ais-netherlands.nl
2. D.P. Bertsekas, J.N. Tsitsiklis, *Neuro-Dynamic Programming* (Athena Scientific, Belmont, 1996)
3. Frontier Information Technologies B.V., http://www.frontier.nl
4. S.P. Galis, M.A. Brouwer, T. Joustra, Optimization of yearly airport capacity within noise limits at Schiphol Airport, in *The 33rd International Congress and Exposition on Noise Control Engineering*, 2004, http://www.schiphol.nl/media/portal/$_$scholieren$_$studenten/pdf/pdf$_$files/noise$_$management$_$v1$_$m56577569830678617.pdf
5. A. Genz, Numerical computation of the multivariate normal probabilities. J. Comput. Graph. Stat. **1**, 141–150 (1992)
6. H.J. Kushner, P. Dupuis, *Numerical Methods for Stochastic Control Problems in Continuous Time* (Springer, New York, 2001)
7. A.M. Law, W.D. Kelton, *Simulation Modeling and Analysis*, 2nd edn. (McGraw-Hill, New York, 1991)
8. Luchthavenverkeerbesluit Schiphol – Besluit van 26 November 2002, tot vaststelling van een luchthavenverkeerbesluit voor de luchthaven Schiphol, http://www.verkeerenwaterstaat.nl/Images/Luchthavenverkeerbesluit2002$_$tcm163-90970$_$tcm195-162956.pdf
9. T.R. Meerburg, Noise load management at Schiphol – a stochastic dynamic approach. MSc Thesis, Applied Mathematics, University of Twente (2006), http://wwwhome.math.utwente.nl/~boucherierj/MeerburgMScthesis.pdf
10. M.L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (Wiley, New York, 1994)
11. R.S. Sutton, A.G. Barto, *Reinforcement Learning* (MIT Press, Cambridge, 1998)
12. The MathWorks, Inc., http://www.mathworks.com
13. H.M.M. van der Wal, P. Vogel, F.J.M. Wubben, Voorschrift voor de berekening van de L_{den} en L_{night} geluidbelasting in dB(A) ten gevolge van vliegverkeer van en naar de luchthaven Schiphol. NLR-CR-2001-372, National Aerospace Laboratory NLR (2001)

Chapter 12
Allocation in a Vertical Rotary Car Park

M. Fackrell and P. Taylor

Abstract We consider a vertical rotary car park consisting of l levels with c parking spaces per level. Cars arrive at the car park according to a Poisson process, and if there are parking spaces available, they are parked according to some allocation rule. If the car park is full, arrivals are lost. Cars remain in the car park for an exponentially distributed length of time, after which they leave. We develop an allocation algorithm that specifies where to allocate a newly-arrived car that minimises the expected cumulative imbalance of the car park. We do this by modelling the working of the car park as a Markov decision process, and deriving an optimal allocation policy. We simulate the operation of some car parks when the policy decision making protocol is used, and compare the results with those observed when a heuristic allocation algorithm is used.

Key words: Rotary car park, Markov decision process

12.1 Introduction

In large modern cities throughout the world, because of the high cost of land and the increasing use of motor vehicles, the problem of where to park cars presents a significant challenge for urban planners. One solution has been to build multi-storey car parks, thereby greatly increasing the number of cars parked per square metre of footprint. However, despite this, valuable space is taken up with access ramps, head space, and lifts and stairs. Also, since drivers park their cars themselves, minor accidents occur and driver safety is an issue. Multi-storey car parks must be

M. Fackrell (✉) • P. Taylor
School of Mathematics and Statistics, University of Melbourne, VIC 3010, Australia
e-mail: fackrell@unimelb.edu.au; taylorpg@unimelb.edu.au

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_12

adequately ventilated to cater for exhaust fumes, and their construction costs are very high since such structures need to be made with reinforced concrete and steel.

Another solution that enables cars to be parked more tightly is an *automated (car) parking system* (*APS*), see Wikipedia [13], [14]. Here, cars are moved into position by the driver or attendant at the entrance to the car park, and are then mechanically moved or stacked into a vacant space. In an *APS* damage to cars and injuries to drivers are virtually eliminated, and ventilation is no longer a problem since cars are not driven into position. Also, construction costs per car are reduced as cars can be packed much more tightly. A case study by Monahan [6] concluded that building a freestanding above grade (ground) multi-storey car park with 203 car spaces and overall surface area 8387 m^3 costs \$29,600 per space, whereas an *APS* with 217 car spaces and overall surface area 5409 m^3 costs \$28,170 per space. Wikipedia [13] suggested that the saving per space is even greater for *APS*s when they are built below building and below grade (ground).

The first known *APS* was built in Paris in 1905 at the Garage Rue de Ponthieu, where a lift moved cars to a level where they were parked by an attendant. From the 1920s onwards the *paternoster* (see Wikipedia [15]) system was used. The paternoster *APS* was much like a vertical ferris wheel where cars were parked in a carriage and then moved upwards so the next carriage was ready to receive another car. A Google search for "vertical rotary car park" brings up links to videos on YouTube (see, for example, [17–21]) and numerous images of such car parks and their variants.

YouTube [18] shows a video of a small vertical rotary car park in operation. *Parkmatic*, the manufacturer, make car parks that cater for seven, eight, ten, or twelve vehicles. In the video we see a man driving onto a turntable in front of the car park's entrance. He then exits his car and a control console enables him to rotate the turntable so that the car is now aligned with the entrance. Next he reenters his vehicle and drives onto the carriage where the car will be parked. He then exits the car, returns to the console and keys in a code, and the carriage moves upwards to allow an empty carriage to be positioned at the entrance ready for the next car. In order to retrieve his car, he enters the same code, and the carriage returns the car to the entrance/exit position taking the shortest distance by rotating either clockwise or anticlockwise. The man then enters his car and backs out onto the turntable and returns to the console to rotate the turntable so the car can be positioned to drive away. There are safety features built into the system including sensors which prohibit the driver entering the car park when the carriages are moving, and an emergency stop button. The system can also be reset from the console. In the video YouTube [19] an animation of a much larger vertical rotary car park in operation is shown. The manufacturer is *SimPark Infrastructure*.

Although the car park by *Revo-park* depicted in the animated video YouTube [17] is not strictly a vertical rotary car park as described above, it is a multi-story mechanised car park that stacks cars. The car park is a cylindrical structure of ten levels where each level can park 50 cars. The levels may be above or below ground. Each level of the car park consists of three concentric sections. The outer circular section, being the largest, is where most of the cars are parked. In the middle section a rotating ring allows cars to be moved horizontally around the car park at the same level. Cars may also be parked in this section. The inner section houses a vertical

conveyer which allows cars to be moved to other levels. *Revo-park* also manufactures rectangular car parks which, at each level, consist of a row of parking spaces, a linear mechanism which moves cars horizontally, and a row of vertical conveyers. Apartment or office blocks can be built around such rectangular car parks. *PTV Vissim* (see *FATA Automation, Inc.*, YouTube [20]) provide animated videos of similar *APS*s.

There are a number of patents describing various *APS*s. Buch [3] proposed a system that consisted of a number levels arranged around a central shaft which contained a helical mechanism that would lift the cars to the appropriate level to be parked. In Beretta [2] the car park had at least one vertical elevator to take the cars to levels, and at least two "trolley" mechanisms to maneuver the car horizontally into a parking space. In Schween [9] the *APS* could park and retrieve multiple vehicles simultaneously. Each level consisted of rails which could move cars in a longitudinal and transverse manner, and a vertical elevator that moves cars between levels. Added features included the videotaping of arriving cars to protect against fraudulent insurance claims, and a control mechanism that measures the entering vehicle's size and rejects those that are over-sized. Trevisani [11] and Vita [12] both proposed *APS*s that had a central shaft with car spaces located radially around it at a number of levels. Vita [12] had an inner ring which could move cars horizontally around a level until a space was found in an outer ring where the car was parked, similar to the *Revo-park* system (YouTube [17]) described above. On the other hand, in Trevisani [11], the car park only had one ring where cars could be parked, the central shaft mechanism rotating to align cars with empty spaces.

In the academic literature there is little concerning the operation of vertical rotary car parks. Gomes [4] developed a model that optimised the utilisation of a parking system (not necessarily a vertical rotary car park) by taking into consideration the stochastic demand for spaces. Hwang and Lee [5] analysed a vertical rotary car park and calculated the expected parking time by treating the car park as an $M/M/n/n$ queue where n was the number of parking levels, each accommodating at most one car.

We note here that for all of these car parking systems, there is no mention of any algorithm or process with which to park cars so that the movement of the vertical conveyers and horizontal platforms is minimised in order to reduce wear and tear on the mechanisms. It appears that cars are merely parked in an available space.

Consideration of how to allocate cars to parking spaces in order to minimise the distance the mechanisms move (or, as in our case below, to minimise the imbalance of the car park) is the main focus of this paper.

We model a vertical rotary car park of the paternoster type with l levels and c cars-per-level with an infinite-horizon Markov decision process (*MDP*). The objective function to be minimised is a measure which combines the imbalance of the car park with the distance it turns in order to accommodate arriving cars. From the *MDP* we calculate a policy which indicates where an arriving car should be parked, given the current configuration of cars, that minimises the objective measure over the (infinite) course of the operation of the car park. After the policies have been calculated for some examples, we simulate the running of the car park and compare the *MDP* allocation algorithm with a heuristic allocation algorithm.

In the next section we describe the vertical rotary car park under consideration and briefly introduce Markov decision processes (*MDP*s). Section 12.3 contains a detailed description of the *MDP* used to model the vertical rotary car park. In Sect. 12.4 we run the *MDP* algorithm and calculate policies for an 8-level car park with 1, 2, and 3-cars-per-level, for various loads on the system and turning penalties. The details of the convergence of the algorithm for these examples are presented in section "*MDP* Convergence Results" in Appendix. Section 12.4 contains an in-depth analysis of the policy for the 8-level, 1-car-per-level car park. Part of the policy is presented in section "Policy When $l = 8$ and $c = 1$" in Appendix. In Sect. 12.4 we simulate the running of an 8-level, 3-cars-per-level car park under various loads with different turning penalties, and two different parking time distributions - one exponential, the other Erlang. We also compare the *MDP* allocation algorithm with a heuristic algorithm. The results and some histograms are presented in section "Simulation Results for $l = 8$ and $c = 3$" in Appendix. We also ran some simulations for a 20 level, 1-car-per-level car park, the results of which are given in section "Simulation Results for $l = 20$ and $c = 1$" in Appendix. The paper concludes in the last section with some ideas for future research.

12.2 Background

12.2.1 The Car Parking Allocation Problem

Consider a vertical rotary car park of l levels where each level can accommodate a maximum of c cars, see Fig. 12.1. Cars arrive randomly according to some arrival process, and park for a random time. If the car park is full, arriving cars are lost to the system, that is, there is no queueing. When a car arrives, if the car park is not full, a space within a level is chosen in which to park the car, the car park is rotated (in a direction that minimises the distance turned) until the chosen level is at the bottom, and the car is driven on. When a car needs to depart, the car park is rotated so that the level where the departing car is situated is at the bottom, and the car is driven off. After each arrival or departure the car park remains static until the next arrival or departure.

During the operation of the car park, when a car leaves, no decision needs to be made as the car park must be turned to allow the car to depart. However, when a car arrives, a decision must be made as to where to park it. Various objectives could be taken into consideration. In this paper we focus on minimising how unbalanced, in some sense, the car park becomes. If the car park is close to being half full but most of the cars are on one side, then this would be considered as "highly unbalanced". However, if cars are placed such that there are approximately the same number on either side, then this would be considered as "almost balanced". If during its operation the car park becomes highly unbalanced quite frequently, then the car park would need to be engineered for this. However, if it rarely becomes highly

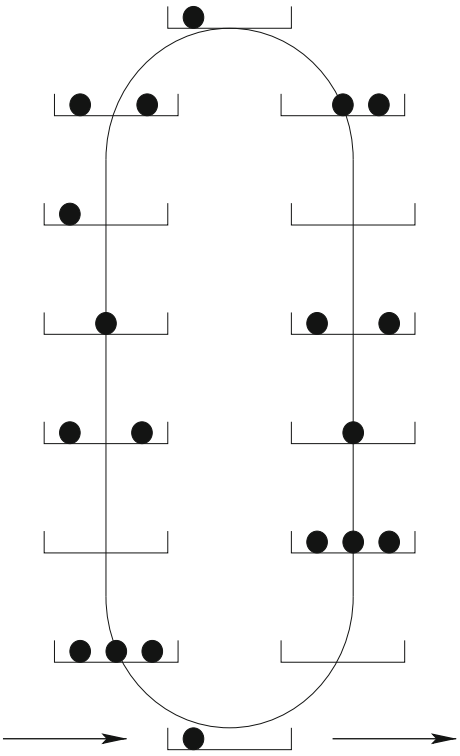


Fig. 12.1: A vertical rotary car park with 14 levels and 3 cars per level. Cars enter from the *left* and exit from the *right*

unbalanced, then the cost of constructing the car park would be conceivably a lot less. It would also seem reasonable to consider minimising how much the car park turns in deciding where an arriving car is to be parked. The more the car park turns, the greater the contribution to its general wear-and-tear and repair costs. Deciding on a measure of imbalance for the car park is somewhat arbitrary. The measure we describe here takes into consideration the turning of the car park by adding a turning penalty proportional to the distance turned, but it is small compared to the effect of imbalance.

For any given configuration ϕ of the car park, the imbalance is simply the absolute value of the difference between the number of cars on each side of the car park. We do not include the bottom or top levels in the calculation. Thus the imbalance of the configuration ϕ is $|\#\phi(L) - \#\phi(R)|$, where $\#\phi(L)$ and $\#\phi(R)$ denote the number of cars on the left and right hand sides of the car park, respectively.

If an l-level, c-cars-per-level car park moves from an initial configuration ϕ_i to a final configuration ϕ_f via the sequence $\phi_i \to \phi_1 \to \phi_2 \to \ldots \to \phi_v \to \phi_f$ (taking v turns), and τ is the turning penalty, the imbalance Θ (of the move) is given by

$$\Theta(\phi_i, \phi_f) = \max_{j \in \{1,2,\ldots,v\}} \left| \#\phi_j(L) - \#\phi_j(R) \right| + v\tau. \tag{12.1}$$

We illustrate the calculation of the imbalance Θ with a simple example. Consider an 8-level, 1-car-per-level car park. Refer to Fig. 12.2. The leftmost diagram is the initial configuration of the car park where the filled in circles represent occupied levels, and the empty circles represent unoccupied levels. In this example, the red car (circle) needs to depart. The dotted vertical line separates the left and right hand sides of the car park, the bottom and top levels being ignored. Initially the imbalance $I = 1$ as there are two cars on the left hand side and one on the right hand side. In the second diagram the car park has been rotated anticlockwise (to minimise the distance the departing car travels to reach the bottom) by one level. The imbalance is calculated as the maximum of any previous imbalances and the absolute value of the current difference in the number of cars on the left and right hand sides, plus a turning penalty, in this case $\tau = 0.001$. Thus, the imbalance is calculated as $I = \max\{1,2\} + 0.001 = 2.001$. In the third diagram the car park has been rotated anticlockwise one level and the imbalance is now calculated as $I = \max\{2,2\} + 0.002 = 2.002$. In the fourth diagram the red car has reached the bottom and the imbalance is calculated as $I = \max\{2,0\} + 0.003 = 2.003$. In the final diagram the red car departs. No more calculations of the imbalance are required as it would be the same as in the fourth diagram. Thus $\Theta(\phi_i, \phi_f) = 2.003$. We note here that the initial configuration (first diagram) would be the final configuration of the previous move. In practice we do not include the initial configuration in the calculation of the imbalance because it has already been used in the calculation of the imbalance of the previous move and we do not want to count it twice.

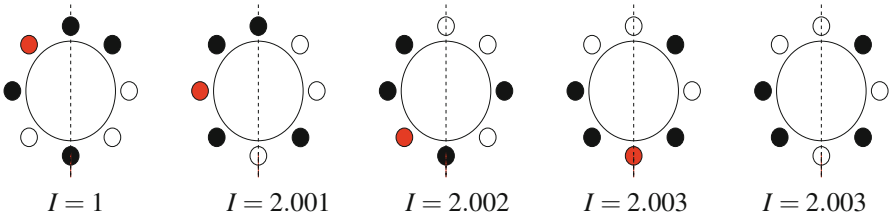


Fig. 12.2: The sequence of configurations of an 8-level, 1-car-per-level car park when the *red car* (*circle*) needs to depart

In an l-level, c-cars-per-level car park, there are $(c+1)^l$ different configurations the car park can be in. We will call the configurations "states" from now on. We can represent each state as a l-digit number in base $c+1$ as follows. The first digit is the number of cars parked in the bottom level, the second digit is the number of cars parked in the second level moving around the car park in a clockwise direction, and so on until the lth digit, which is the number of cars parked in the level immediately to the right of the bottom level. For example, the state of the system in the first diagram in Fig. 12.2 is represented by the 8-digit binary number 00111101, or by the decimal number 61. The state of the car park in Fig. 12.1 is represented by the 14-digit base 4 number 03120212112031, or by the decimal number 56780173.

We note here that the position of each car in a level is not taken into consideration when calculating the state's label. If in the example depicted in Fig. 12.1, we wished to take into consideration the position of cars in each level, then we would effectively need to consider a 42-level, 1-car-per-level car park.

The main focus of this paper is to develop a policy that tells us where to park an arriving car so that the overall expected rate of increase of imbalance measure (which may include a turning penalty) is minimised over the course of the operation of the car park. We need to take into account that cars are arriving according to some random arrival process, and that cars will leave once they have completed a random time in the car park.

For example, if an 8-level, 1-car-per-level car park is in the state depicted in the first diagram of Fig. 12.3, that is, state 73, where do we place an arriving car? Three possibilities are to states 83, 147, and 165 (the arriving car is in red), incurring imbalance measures (with $\tau = 0.001$) of 1.002, 0.001, and 1.001, respectively. At first glance it would appear that state 147 is the preferred option since the imbalance is the least. However, given that cars are arriving and leaving randomly we need to consider what will happen next. That is, we need to place an arriving car so that the imbalance is minimised over the course of the whole operation of the car park, not just where the imbalance would be minimised for the next step only.

It turns out, as we shall see later, that each of the three options can be optimal, depending on the load (the ratio of the arrival rate to the service rate) on the system. When the load on the system is small, state 147 is optimal. When the load is around 1/2, state 165 is optimal. And when the load is 1 or more, state 83 is optimal. Indeed, for example, when in state 147, an arrival placed in one of the four available spaces incurs an imbalance of 2.002 or 2.003, but a departure incurs an imbalance of 0, 1.001, or 2.004. So, when the next event is more likely to be a departure than an arrival, the expected imbalance over two steps is minimised when an arriving car finding the car park in state 73, should be parked to change the state to 147. When the load is around 1/2 arrivals are more likely to occur (but still less likely than a departure), and state 83 should be chosen. In heavier traffic (load ≥ 1) when arrivals are even more likely than departures, state 165 should be chosen.

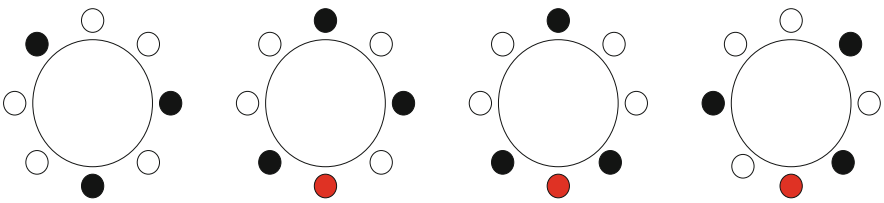


Fig. 12.3

12.2.2 Markov Decision Processes

In a *Markov decision process* (*MDP*), for each state of the system, a decision needs to be made as to which of several *actions* is taken. When the state changes, the state transition probabilities and the reward received (or cost incurred) in taking the action, depend on both the state and the action. The objective is to determine a policy (by finding which action should be taken for each state of the system) to maximise (minimise) the *expected* reward (cost) over a time period which could be finite or infinite.

We now formally define an *MDP*. The operation of a system is described by a homogeneous, finite-state, discrete-time stochastic process $X(t), t = 1, 2, \ldots$. Let the state space be $S = \{1, 2, \ldots, m\}$. For each $i \in S$, there is a set of actions $A(i)$, one of which will need to be decided on at each time step. The transition probabilities between states depend on the action, that is, for $i, j \in S$,

$$p(j|i,a) \;=\; P\left(X(t) = j | X(t-1) = i, a \in A(i)\right). \tag{12.2}$$

Associated with each state transition $i \to j$ and action $a \in A(i)$, is a reward $r(j|i,a)$. Then, the expected reward in a single transition from state i when action $a \in A(i)$ is taken, is

$$r(i,a) \;=\; \sum_{j=1}^{m} p(j|i,a) r(j|i,a). \tag{12.3}$$

The objective is to determine, for each state, an action (decision) that will maximise the expected long-term reward accumulated from the start of the process until the end. The whole set of actions for each state and time point is called a *policy*. We assume for the time being that the time horizon is finite, that is the process finishes at some time T in the future.

For $i \in S$, $t = 1, 2, \ldots$, let $V_t(i)$ be the maximum expected reward that can be accrued from time t until time T, given that the state of the system at time t is i. Then the $V_t(i)$ satisfy Bellman's equation

$$V_t(i) \;=\; \max_{a \in A(i)} \left\{ r(i,a) + \sum_{j=1}^{m} p(j|i,a) V_{t-1}(j) \right\}, \tag{12.4}$$

see Taha [10] or Winston [16]. We note here that our formulation of Bellman's equation is slightly different. We index the time by the number of periods t left until time T, which means we write $V_t(i)$ in terms of the $V_{t-1}(j)$. For $i \in S$, Eq. (12.4) can be solved recursively by letting, $V_0(i)$ be a fixed value function. We note that in the above explanation we can replace "reward" with "cost" and consequently, replace "maximise" with "minimise".

12.3 The Markov Decision Process

For the l-level, c-cars-per-level vertical rotary car park we wish to determine a policy so that the expected cumulative imbalance (incorporating the turning penalty) is minimised over the course of the car park's operation. Cars arrive according to a Poisson process with parameter λ, and spend an exponentially distributed length of time with parameter μ in the car park before departing.

The state space is $S = \{1, 2, \ldots, (c+1)^l\}$. Let ϕ_t be the state of the system (that is, the configuration of the parked cars) after event (arrival or departure) $T - t$ has occurred. Let $\Theta(\phi_t, \phi_{t-1})$ be the imbalance incurred when moving from state ϕ_t to state ϕ_{t-1}. Denote by $A(\phi_t)$ the set of all states accessible from state ϕ_t if an arrival occurs. Let $d_s(\phi_t)$ denote the state that is moved to if a car departs from space s when the car park is in state ϕ_t. The load on the car park is $\rho = \lambda/\mu lc$. Any car arriving to find the car park full is lost to the system, that is, there is no queueing.

We assume that the car park operates over a finite time horizon, $0, 1, \ldots, T$. Let $V_t(\phi_t)$ be the minimum expected cumulative imbalance from event $T - t$ to T if the car park is in state ϕ_t. If the car park is in state ϕ_t, the probability that the next event is an arrival is $\lambda/(\lambda + \mu|\phi_t|)$, where $|\phi_t|$ is the number of cars when the car park is in state ϕ_t. If an arrival occurs we need to decide which state in $A(\phi_t)$ to move to. The probability that the next event is a departure from space s is $\mu/(\lambda + \mu|\phi_t|)$.

Now, for $t = 1, 2, \ldots$,

$$V_t(\phi_t) = \frac{\lambda}{\lambda + \mu|\phi_t|} \min_{\phi_{t-1} \in A(\phi_t)} \{V_{t-1}(\phi_{t-1}) + \Theta(\phi_t, \phi_{t-1})\} + \sum_s \frac{\mu}{\lambda + \mu|\phi_t|} \left(V_{t-1}(d_s(\phi_t)) + \Theta(\phi_t, d_s(\phi_t))\right). \tag{12.5}$$

To find the optimal policy we set, for all $\phi_0 \in S$, $V_0(\phi_0) = 0$, and recursively calculate $V_t(\phi_t)$ until the policy no longer changes. Essentially, we determine the policy for an infinite horizon *MDP* as the number of recursions is not specified, see Altman [1], Sect. 15.2.

In practice, since for each $\phi \in S$, $V_t(\phi)$ increases as t increases, we use as our stopping criterion

$$\max_{\phi \in S} \left| \frac{V_t(\phi)}{t} - \frac{V_{t-1}(\phi)}{t-1} \right| < \varepsilon, \tag{12.6}$$

where $\varepsilon > 0$ is some predetermined tolerance. The algorithm converges, see Odoni [7] or Altman [1], Sect. 15.2.

12.4 Numerical Results

We ran the *MDP* algorithm with $l = 8$ and $c = 1, 2, 3$, for $\rho = 0.1, 0.5, 1, 2$ and $\tau = 0.001, 0.01, 0.1$ (Table 12.1). In practice, we observed that if the policy did not change from one iteration to the next, then the algorithm had not necessarily converged. Since the stopping criterion (12.6) depends only on the values of $V_t(\cdot)$ and

the policy is discrete, it is possible that it can remain unchanged for a few iterations before changing. However, as the number of iterations increased, the runs of no change in the policy increased in length, and if there were any changes, they occurred in a decreasing number of places. In Table 12.2 in section "*MDP* Convergence Results" in Appendix we have tabulated the number of iterations the algorithm took to converge in each case (using $\varepsilon = 0.01$ for $l = 1$ and $\varepsilon = 0.005$ for $l = 2, 3$), and also reported the last iteration when a change in policy occurred. The run times were also recorded.

As expected the number of iterations needed for convergence (and corresponding run-times) increased as c increased. For $c = 1$, the number of iterations for convergence was least when $\rho = 2$ and greatest when $\rho = 0.5$. The number of iterations required for convergence when $\rho = 1$ was greater than that when $\rho = 0.1$. A similar trend was observed for $c = 2, 3$ except that the number of iterations required for convergence when $\rho = 0.1$ was greater than that when $\rho = 1$. These trends were all independent of τ. However, for $c = 1, 2, 3$, when $\tau = 0.1$, there was a marked increase in the number of iterations required for convergence. The highest run-time was nearly 10 h for 156 iterations (when $c = 3$, $\rho = 0.5$, and $\tau = 0.1$).

We now discuss some of our findings. It is impossible to present all of the policies calculated for each example as the state spaces are too large. Instead we shall focus on some observations made when $c = 1$, the simplest case. In Tables 12.3 and 12.4 in section "Policy When $l = 8$ and $c = 1$" in Appendix we have given the portion of the policy calculated for $\phi = 65, 66, \ldots, 109$. For given values of ρ and τ, the entry in the row corresponding to state ϕ gives the optimal next state when an arrival occurs and the car park is in state ϕ.

We first note that the policy is independent of ρ and τ for 37 (out of 45) states. For two of the remaining eight states the policy is independent of τ. Next we will comment on the policy for some specific states.

Consider, for example, state 108, which is depicted in Fig. 12.4. The policy stipulates that the car should be parked in the bottom space so the car park moves to state 109 incurring no imbalance, no matter what the values of ρ and τ are. To see why this may be the case, observe that when the car park is in state 108, when a car arrives, it can move to any of states 55, 109, 199, or 217, incurring imbalances of $1 + \tau$, 0, $2 + 4\tau$, or $1 + \tau$, respectively. The policy chooses the state that minimises the imbalance over one time-step. This would seem to be sensible as the minimum imbalance is zero and all the others are greater than one.

State 109 is depicted in Fig. 12.5. Here the policy stipulates that an arriving car be parked so that the car park moves to either state 183 or 219, independent of ρ and τ. When a car arrives, the car park can move to any of states 183, 215, and 219 incurring imbalances of τ, $1 + 4\tau$, and τ, respectively. The policy chooses states 183 and 219 over state 215 as the one time-step imbalance is a lot less. From the diagram we can see that because of symmetry neither of the states 183 and 219 are favored over each other. In practice, such a tie can be broken by randomly selecting one of the two states.

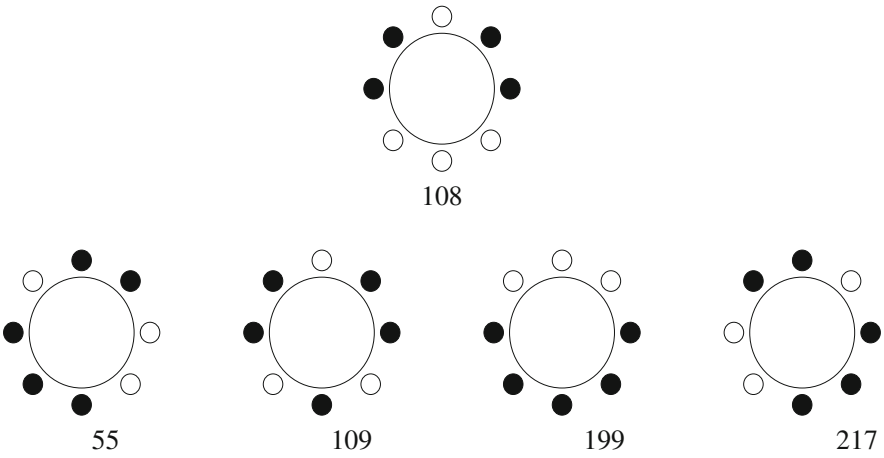


Fig. 12.4: State 108 can move to any of states 55, 109, 199, and 217 when an arrival occurs

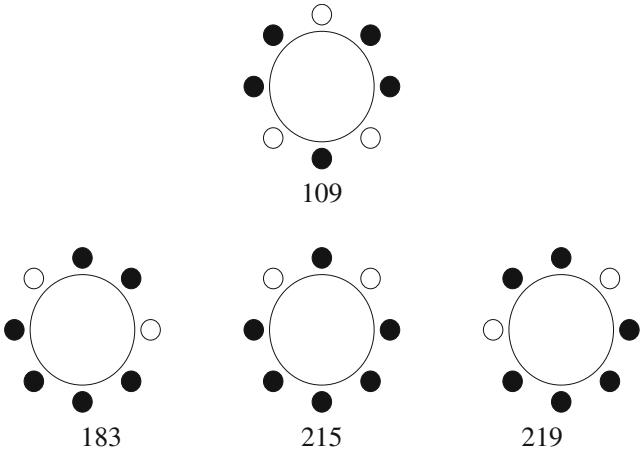


Fig. 12.5: State 109 can move to any of states 183, 215, and 219 when an arrival occurs

For state 67, the policy stipulates that an arriving car should be parked so that the car park moves to state 53, independent of ρ and τ, see Fig. 12.6. From state 67 the car park can move to any of states 27, 53, 105, 135, and 209, incurring imbalances of $2+3\tau$, $2+4\tau$, $2+3\tau$, $1+\tau$, and $2+2\tau$, respectively. Unlike the previous two examples, the policy chooses the state with the maximum imbalance over one time-step. This seems counterintuitive until we see what possible imbalances occur in moving from state 53 compared with the other four states.

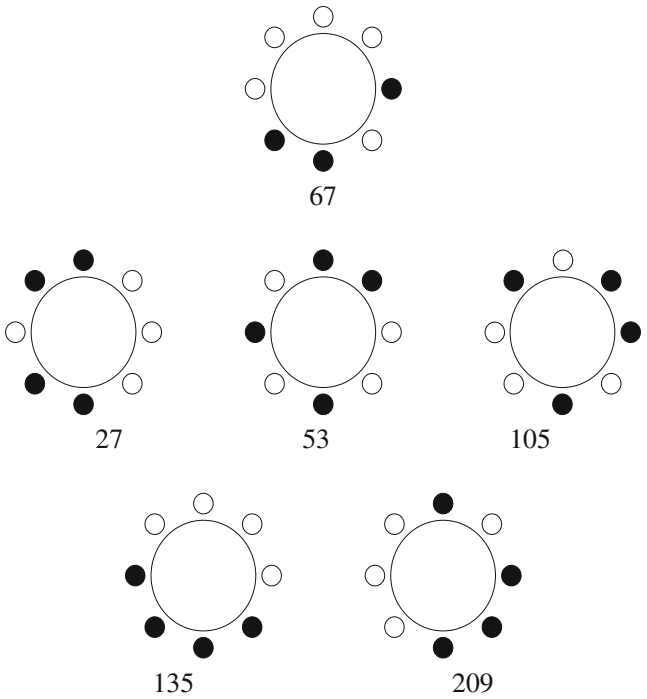


Fig. 12.6: State 67 can move to any of states 23, 53, 105, 135, and 209 when an arrival occurs

In Table 12.1 all the states that can be reached from states 27, 53, 105, 135, and 209, and their corresponding one time-step imbalances, are listed. Even states indicate that a departure has occurred, and odd states indicate that an arrival has occurred. The two time-step imbalances from state 67 are also given. We can see that the imbalances from 67 to 53 to ϕ are collectively less than for any other state. The sum of all imbalances is $22+48\tau$ which is smaller than that of the other states ($26+40\tau$, $24+40\tau$, $28+24\tau$, and $30+28\tau$ for states 27, 105, 135, and 209, respectively). So from state 67, the policy chooses state 53 because its two time-step imbalance is least in total.

12.5 Simulation Results

We simulated the working of the car park when $l = 8$ and $c = 3$ using the *MDP* policy to allocate arriving cars. We considered all cases, that is, when $\rho = 0.1, 0.5, 1, 2$ and $\tau = 0.001, 0.01, 0.1$. We first considered a Poisson arrival stream and an exponential service time distribution, and then considered a 4-phase Erlang service time distribution. For each simulation we started with the car park empty and started

$27 \to \phi$	26	55	98	109	140	178	199	217
$\Theta(27,\phi)$	2	$1+\tau$	$1+3\tau$	$1+2\tau$	$1+\tau$	$2+4\tau$	$1+2\tau$	$1+3\tau$
$\Theta(67,27)+\Theta(27,\phi)$	$4+3\tau$	$3+4\tau$	$3+6\tau$	$3+5\tau$	$3+4\tau$	$4+7\tau$	$3+5\tau$	$3+6\tau$
$53 \to \phi$	52	76	82	107	155	167	168	213
$\Theta(53,\phi)$	0	$1+2\tau$	$1+4\tau$	τ	$1+\tau$	$1+3\tau$	$1+3\tau$	$1+2\tau$
$\Theta(67,53)+\Theta(53,\phi)$	$2+4\tau$	$3+6\tau$	$3+8\tau$	$2+5\tau$	$3+5\tau$	$3+7\tau$	$3+7\tau$	$3+6\tau$
$105 \to \phi$	44	74	91	104	151	164	181	211
$\Theta(105,\phi)$	$1+3\tau$	$1+3\tau$	$1+2\tau$	1	$1+4\tau$	$1+2\tau$	$1+\tau$	$1+\tau$
$\Theta(67,105)+\Theta(53,\phi)$	$3+6\tau$	$3+6\tau$	$3+5\tau$	$3+3\tau$	$3+7\tau$	$3+5\tau$	$3+4\tau$	$3+4\tau$
$135 \to \phi$	14	31	61	121	134	194	224	241
$\Theta(135,\phi)$	$3+\tau$	$3+2\tau$	$3+3\tau$	$3+4\tau$	1	$1+\tau$	$3+2\tau$	$3+3\tau$
$\Theta(67,135)+\Theta(53,\phi)$	$4+2\tau$	$4+3\tau$	$4+4\tau$	$4+5\tau$	$2+\tau$	$2+2\tau$	$4+3\tau$	$4+4\tau$
$209 \to \phi$	28	59	70	117	143	162	208	233
$\Theta(209,\phi)$	$2+4\tau$	$2+3\tau$	$1+2\tau$	$2+2\tau$	$2+3\tau$	$1+\tau$	2	$2+\tau$
$\Theta(67,135)+\Theta(53,\phi)$	$4+6\tau$	$4+5\tau$	$3+4\tau$	$4+4\tau$	$4+5\tau$	$3+3\tau$	$4+2\tau$	$4+3\tau$

Table 12.1: For turning penalty τ, the one time-step imbalances in moving from states 27, 53, 105, 135, and 209 when either an arrival or a departure occurs, and the respective two time-step imbalances in moving from state 67

recording the imbalance and distance moved after 500,000 events (arrivals and departures) for a further 500,000 events. The time taken for the simulations was around 5 min.

In order to assess further the *MDP* policy allocation algorithm we repeated the simulations using a heuristic method as follows. If an arriving car found a space in the bottom level vacant it was placed there. If, however, an arriving car found the bottom level full, it was allocated a vacant space elsewhere in the car park randomly. This last vacated space/random allocation algorithm is abbreviated to *LVS*.

For each simulation the car park was never full when $\rho = 0.1$, almost never full when $\rho = 0.5$, full about 15% of the time when $\rho = 1$, and full about 52% of the time when $\rho = 2$.

In Tables 12.5 and 12.6 in section "Simulation Results for $l = 8$ and $c = 3$" in Appendix, with the occupancy distribution taken to be exponential and Erlang, respectively, we recorded the average and maximum imbalance (for a complete move, see Eq. (12.1)), and the average distance the car park moved, over the 500,000 recorded events, for the *MDP* and *LVS* allocation algorithms. In every case, the maximum distance the car park moved was 4. Furthermore, for $\tau = 0.001$ histograms for the distribution of imbalance and distance moved are presented in Figs. 12.7, 12.8, 12.9, 12.10.

When the optimal policy was used, and the service time distribution was exponential the average imbalance and average distance moved were always lower than when the service time distribution was Erlang, although both measures were close. This seems plausible because an exponential service time was assumed when calculating the *MDP* policy. The same trend was observed, most of the time, with the maximum imbalance.

The average imbalance was considerably lower for the *MDP* policy algorithm compared to the *LVS* algorithm, with the differences becoming less pronounced as ρ increased. This trend was observed for both the exponential and Erlang service time distributions. A similar trend was observed when considering the maximum imbalance. Again, this seems plausible as the *MDP* policy is optimal when the imbalance measure was minimised.

For each value of ρ, for both the exponential and Erlang cases, the average imbalance increased as τ increased, except when $\rho = 0.1$ and $\tau = 0.1$. This general trend was expected since with increasing τ the imbalance increased [as the imbalance measure incorporated the turning penalty, see Eq. (12.1)]. It was not clear, however, why the trend was reversed when $\rho = 0.1$ and $\tau = 0.1$. The trend was not as pronounced for the maximum imbalance, particularly when the exponential service time was used.

For each value of τ, for both the exponential and Erlang cases, the average imbalance was least when $\rho = 2$. We expected this because when the car park was full, or nearly full, most of the time, the imbalance would be zero, or close to zero. The next smallest average imbalance was for $\rho = 0.1$. Again, we expected this because when the car park was not very full at lot of the time, the imbalance would be low. The average imbalance for $\rho = 0.5$ was always higher than when $\rho = 1$, but not by much. When $\rho = 0.5, 1$ the car park was more likely to become "lop-sided", hence the higher average imbalances.

For both exponential and Erlang service time distributions, the average distance moved when the *MDP* algorithm was used was greater than when the *LVS* algorithm was used, except when $\rho = 2$ and $\tau = 0.001, 0.01, 0.1$, and $\rho = 1$ and $\tau = 0.1$. When the load on the system was $\rho = 0.1, 0.5, 1$ the car park was empty most of the time so when using the *LVS* algorithm, an arriving car was usually placed at the bottom of the car park with no turning. But when $\rho = 2$ the car park was full about half of the time, so arriving cars needed to be placed elsewhere. Since the *MDP* algorithm has an turning penalty built into the imbalance measure that is minimised, the average distance moved was less than that when the *LVS* algorithm was used, albeit only marginally.

For each value of ρ, for both the exponential and Erlang cases, the average distance decreased with increasing ρ except for the Erlang case when $\rho = 2$ and $\tau = 0.01$, but the increase here was very small. This general trend was to be expected as turning the car park was penalised more as τ increased.

When the *MDP* algorithm was used, in the Erlang case, for each value of τ, the average distance increased with decreasing ρ. This would appear to make sense because the less busy the car park is, the higher the likelihood of parking a car "on the opposite side" to minimise imbalance, at least in one time-step, resulting in higher distances. However, for the exponential case, apart from the lowest average distance achieved when $\rho = 2$ the trend is not the same as with the Erlang case. It is not clear why this was the case.

For $\tau = 0.001$ and each value of ρ, the histograms of the imbalance and distance are displayed in Figs. 12.7, 12.8, 12.9, 12.10. Histograms are given for the *MDP* and *LVS* allocation algorithms and for when the service time distribution is exponential and Erlang.

The first thing to notice is that, when the exponential and Erlang service time distributions were used, the histograms are similar for both imbalance and distance. Thus, we will only comment on the distributions for the exponential case.

For imbalance, for all values of ρ, all histograms were skewed to the right with those for the *LVS* algorithm skewed further to the right. This indicates that smaller imbalances are more likely than large imbalances, and imbalances are generally lower when the *MDP* algorithm was used. For the *MDP* case, the modal imbalance was 0 when $\rho = 0.1, 2$ and 1 when $\rho = 0.5, 1$. However, for the *LVS* case, the modal imbalance was 0 when $\rho = 2$ and 1 when $\rho = 0.1, 0.5, 1$. It is clear then, that the *MDP* allocation algorithm is better than the *LVS* one when minimising the imbalance measure. This conclusion was also drawn when we discussed the average and maximum imbalances previously.

For distance, all histograms, except for *MDP* when $\rho = 0.1$, are skewed to the right. The modal distance was 0 in all cases. Except for *MDP* when $\rho = 0.1, 0.5$, distances of $1, 2$, and 3 were recorded roughly half as many times as a distance of 0, and distances of 4 were roughly half again. For the *MDP* case when $\rho = 0.1$ the distribution was U-shaped with a distance of 0 more likely than 4. Here, it would appear that when the traffic is light, parking "on the other side" of the car park helps minimise the imbalance measure.

The *MDP* algorithm was run when $l = 20$ and $c = 1$ for $\tau = 0.001$ and $\rho = 0.1, 0.5, 1, 2$. The motivation for this exercise was to see how far the *MDP* policy deciding algorithm could be taken. In each case the algorithm took some 30 days and more than 100 iterations to converge! This highlights the difficulty in using the *MDP* approach for larger car parks. As with the smaller car parks ($l = 8$, $c = 1, 2, 3$), for $\tau = 0.001$, and $\rho = 0.1, 0.5, 1, 2$, 500,000 events were simulated for both exponential and Erlang service time distributions. The times to run these simulations ranged from 5 to 10 min, so there is no difficulty in applying the *MDP* allocation algorithm once it has been calculated. The results were compared with the *LVS* algorithm. In section "Simulation Results for $l = 20$ and $c = 1$" in Appendix, Tables 12.7 and 12.8 list the average and maximum imbalance and average distance (the maximum distance was 10 in all cases) for each case. In Figs. 12.11, 12.12, 12.13, 12.14 histograms for the imbalance and distance are shown for both *MDP* and *LVS*, for both the exponential and Erlang cases.

The same trends in the average and maximum imbalance were observed when $l = 8$ and $c = 3$ were seen when $l = 20$ and $c = 1$. The values for the average and maximum imbalance are slightly higher when $l = 20$ and $c = 1$ for both *MDP* and *LVS*, and for exponential and Erlang (the maximum imbalance with *LVS*, Erlang when $\rho = 2$ is the only exception). For the average distance the trends were the same but the values were all higher for $l = 20$ and $c = 1$ with a maximum distance of 10 in all cases. This of course is expected as we have the potential turn through more levels.

12.6 Conclusion

From our examples it is apparent that using the *MDP* allocation algorithm to determine where arriving cars are parked is considerably superior to the heuristic *LVS* algorithm, at least when imbalance is minimised. The *LVS* algorithm outperforms the *MDP* in most cases in minimising the distance turned, but not by much. This could be remedied by increasing the turning penalty τ. Indeed, when $\tau = 0.1$ and $\rho = 2$, *MDP* outperformed *LVS* in minimising distance in all cases. Choosing this option depends on how much distance is valued compared to imbalance.

The main problem with *MDP* is the size of the state space. For $l = 8$ and $c = 3$ the size of the state space is 65,336, and for $l = 20$ and $c = 1$ it is 1,048,576. Computing the optimal policy where the state space has more than one million states would be prohibitive. One possible remedy would be to consider "clumping" levels into "super-levels". For example, if we needed to determine a policy for a 40-level, 3-cars-per-level car park (1.2089×10^{24} states) we could consider it as an 8-level, 15-cars-per-level car park (4,294,967,296 states). This is still much too large and a lot of detail has been lost in clumping 5 3-car levels into one 15-car level. The problem here is which level within the super-level is the car to be parked to minimise the imbalance measure. For the 40-level, 3-cars-per-level car park, the only computationally realistic car park would be a 4-level, 30-cars-per-level one (923,521 states) but the level of detail is low. We will be deciding which quarter of the car park to park arriving cars in.

Another approach to circumvent the large state spaces is to use a heuristic allocation algorithm (for example *LVS*) to determine, via simulation, which states occur most frequently. Then the *MDP* policy could only be calculated for these most frequent states. Then, if when simulating the running of the car park a non-frequent state occurs, the heuristic (*LVS*) algorithm could be used. One problem here though, is the size of the simulation required to determine the most frequent states given that there are a huge number of states that can be realised in larger car parks.

Another approach to cope with the large state space is to use *approximate dynamic programming*, see Powell [8], but this is left for future research.

Throughout this paper we have assumed that any car finding the car park full does not queue and is lost to the system. Allowing a queue to form and having cars enter the car park once a space becomes available on a first-come-first-served basis would make the situation more realistic, especially if cars were arriving and leaving frequently. The analysis would not be too much more complicated because if a car leaves, and if there is a queue, the car first in line will simply replace the one that has just left, and so on until the queue is empty. Performance measures such as the mean queue length and mean waiting time in the queue could be calculated by considering the car park (with queueing) as an $M/M/lc$ queue, or an $M/M/lc/k$ queue (with $k > lc$) if there was a limit to the number of cars that could be queued.

Lastly, it is conceivable that the car park could be modelled from say, being empty at time zero to some finite time in the future. Here we would need a finite horizon *MDP* to model the car park and calculate a time dependent policy. Such a model would be appropriate if the car park was used, for example, by people attending

some event (such as a film or sporting event) where the car park is empty before the event, fills up as people arrive, and then empties after the event. This is left for future research.

Acknowledgements Peter G. Taylor's research is supported by the Australian Research Council (ARC) Laureate Fellowship FL130100039 and the ARC Centre of Excellence for the Mathematical and Statistical Frontiers (ACEMS) CE140100049.

Appendix

Notation

l	Number of levels in the car park
c	Number of cars per level
ϕ	A configuration of the car park
$\#\phi(L)$	Numbers of cars on the left hand side of the car park when in configuration ϕ
$\#\phi(R)$	Numbers of cars on the right hand side of the car park when in configuration ϕ
v	Number of turns
Θ	Imbalance measure
τ	Turning penalty
I	Imbalance
$X(t)$	A homogeneous, finite-state, discrete-time stochastic process
S	A state space
m	Number of states in the state space
i, j	States in the state space
$A(i)$	Set of actions that can be taken in state i
a	An action
$p(j\|i,a)$	Probability of moving from state i to state j if action a is taken
$r(j\|i,a)$	Reward received in moving from state i to state j if action a is taken
$r(i,a)$	Expected reward received in moving from state i if action a is taken
$V_t(i)$	Maximum expected reward that can be accrued from time t until time step T, given that the state of the system at time step t is i

λ	Cars' Poisson arrival rate
μ	Parameter for the exponentially distributed car parking time
t	The number of events (arrivals and departures)
T	The event number at which the process ceases
ϕ_t	State of the car park after t events
$\lvert\phi_t\rvert$	Number of cars in the car park after t events
ρ	Load on the car park
$A(\phi_t)$	Set of all states accessible from state ϕ_t if an arrival occurs
$d_s(\phi_t)$	State that is moved to if a car departs from space s when the car park is in state ϕ_t
$V_t(\phi_t)$	Minimum expected cumulative imbalance from event $T-t$ to T if the car park is in state ϕ_t
ε	Predetermined tolerance level

MDP *Convergence Results*

See Table 12.2.

τ	ρ	Iterations	Run-time
0.001	0.1	5/57	39
	0.5	14/73	50
	1	9/65	45
	2	9/48	34
0.01	0.1	13/58	41
	0.5	13/74	51
	1	12/66	45
	2	12/49	34
0.1	0.1	6/69	48
	0.5	10/88	61
	1	6/80	56
	2	14/60	42

τ	ρ	Iterations	Run-time
0.001	0.01	55/108	2326
	0.5	84/147	3143
	1	35/104	2221
	2	29/62	1240
0.01	0.1	35/111	2334
	0.5	106/150	3038
	1	33/106	2197
	2	31/63	1326
0.1	0.1	35/137	2789
	0.5	73/178	3617
	1	41/128	2598
	2	30/77	1584

τ	ρ	Iterations	Run-time
0.001	0.1	83/120	26,881
	0.5	102/129	29,574
	1	56/78	17,791
	2	41/44	10,115
0.01	0.1	78/123	28,082
	0.5	115/131	29,418
	1	51/80	18,285
	2	39/45	10,115
0.1	0.1	110/152	34,021
	0.5	111/156	35,676
	1	65/97	21,992
	2	40/55	13,232

Table 12.2: The number of iterations and run-time (in seconds) the *MDP* algorithm took for $\tau = 0.001, 0.01, 0.1$ and $\rho = 0.1, 0.5, 1, 2$ when $l = 8$ and $c = 1, 2, 3$ (tables ordered left to right, top to bottom). The first entry in the iterations column is the number of iterations after which the policy ceased to change, and the second entry is the number of iterations for convergence using a tolerance of $\varepsilon = 0.01$ when $c = 1$, and $\varepsilon = 0.005$ when $c = 2, 3$. The run-time is for a single run but is deemed to be typical

Policy When $l = 8$ *and* $c = 1$

See Tables 12.3 and 12.4.

τ	0.001	0.001	0.001	0.001	0.01	0.01	0.01	0.01	0.1	0.1	0.1	0.1
ρ	0.1	0.5	1	2	0.1	0.5	1	2	0.1	0.5	1	2
65	131	41	41	41	131	41	41	41	131	41	41	81
66	145	145	145	145	145	145	145	145	145	145	145	145
67	53	53	53	53	53	53	53	53	53	53	53	53
68	35,137	35,137	35,137	69	35,137	35,137	35,137	69	35,137	35,137	69	69
69	85	85	85	85	85	85	85	85	85	85	85	85
70	51	51	51	51	51	51	51	51	51	51	51	51
71	117	117	117	117	59	117	117	59	59	59	59	59
72	73	73	73	73	73	73	73	73	73	73	73	73
73	147	165	83	83	147	165	83	83	147	165	165	147
74	149	149	149	149	149	149	149	149	149	149	149	149
75	91	91	91	91	91	91	91	91	91	91	91	91
76	153	153	153	153	153	153	153	153	153	153	153	153
77	107	213	213	213	107	213	155	155	107	155	155	155
78	157	157	157	157	157	157	157	157	157	157	157	157
79	123	123	123	123	123	123	123	123	123	123	123	123
80	41	41	41	41	41	41	41	41	41	41	41	41
81	85	85	85	85	85	85	85	85	85	85	85	85
82	83	83	83	83	83	83	83	83	83	83	83	83
83	107	213	213	213	107	213	213	213	107	213	213	213
84	85	85	85	85	85	85	85	85	85	85	85	85
85	171	171	171	171	171	171	171	171	171	171	171	171
86	173	173	173	173	173	173	173	173	173	173	173	173
87	187	187	187	187	187	187	187	187	187	187	187	187
88	89	89	89	89	89	89	89	89	89	89	89	89
89	173	173	173	173	173	173	173	173	173	173	173	173

Table 12.3: A portion of the policy ($\phi = 65, \ldots, 89$) when $l = 8, c = 1$, $\tau = 0.001, 0.01, 0.1$, and $\rho = 0.1, 0.5, 1, 2$

τ	0.001	0.001	0.001	0.001	0.01	0.01	0.01	0.01	0.1	0.1	0.1	0.1
ρ	0.1	0.5	1	2	0.1	0.5	1	2	0.1	0.5	1	2
90	91	91	91	91	91	91	91	91	91	91	91	91
91	183	183	183	183	183	183	183	183	183	183	183	183
92	93	93	93	93	93	93	93	93	93	93	93	93
93	187	187	187	187	187	187	187	187	187	187	187	187
94	189	189	189	189	189	189	189	189	189	189	189	189
95	191	191	191	191	191	191	191	191	191	191	191	191
96	49	49	49	49	49	49	49	49	49	49	49	49
97	89	89	89	89	89	89	89	89	89	89	89	89
98	153	153	153	153	153	153	153	153	153	153	153	153
99	109	109	109	217	109	109	109	217	109	109	109	217
100	51	51	51	51	51	51	51	51	51	51	51	51
101	173	87	87	87	173	87	179	179	173	179	179	179
102	103	103	103	103	103	103	103	103	103	103	103	103
103	119	119	119	119	119	119	119	119	119	119	119	119
104	53	53	53	53	53	53	53	53	53	53	53	53
105	181	181	181	181	181	181	181	181	181	181	181	181
106	107	107	107	107	107	107	107	107	107	107	107	107
107	219	219	219	219	219	219	219	219	219	219	219	219
108	109	109	109	109	109	109	109	109	109	109	109	109
109	183, 219	183, 219	183, 219	183, 219	183, 219	183, 219	183, 219	183, 219	183, 219	183, 219	183, 219	183, 219

Table 12.4: A portion of the policy ($\phi = 90, \ldots, 109$) when $l = 8, c = 1$, $\tau = 0.001, 0.01, 0.1$, and $\rho = 0.1, 0.5, 1, 2$

Simulation Results for $l = 8$ *and* $c = 3$

See Tables 12.5 and 12.6.

	MDP	*MDP*	*LVS*	*LVS*
ρ	Ave/max imbalance	Ave distance	Ave/max imbalance	Ave distance
0.1	0.5578/4.004	1.5210	1.1808/6.004	1.3166
0.5	0.9635/8.004	1.5308	1.9888/9.004	1.5120
1	0.8250/7.004	1.4665	1.1282/8.004	1.4870
2	0.3328/6.002	1.3213	0.3583/7.004	1.3440

	MDP	*MDP*	*LVS*	*LVS*
ρ	Ave/max imbalance	Ave distance	Ave/max imbalance	Ave distance
0.1	0.5693/4.04	1.5208	1.1923/6.04	1.3166
0.5	0.9741/7.04	1.5133	2.0023/9.04	1.5120
1	0.8355/7.04	1.4589	1.1389/8.04	1.4870
2	0.3377/6.03	1.3174	0.3629/7.04	1.3440

	MDP	*MDP*	*LVS*	*LVS*
ρ	ave/max imbalance	ave distance	ave/max imbalance	ave distance
0.1	0.5084/5.4	1.3530	1.3074/6.4	1.3166
0.5	1.0942/7.4	1.4760	2.1375/9.4	1.5120
1	0.9397/7.4	1.4407	1.2460/8.4	1.4870
2	0.3830/6.3	1.3158	0.4091/7.4	1.3440

Table 12.5: The average and maximum imbalance, and maximum distance, recorded for a simulation run of 500,000 events (arrivals and departures) when $l = 8$, $c = 3$, and $\tau = 0.001, 0.01, 0.1$ (tables presented in order). The *MDP* policy and the last vacated/random space (*LVS*) allocation algorithms were used. The service time distribution was exponential

	MDP	*MDP*	*LVS*	*LVS*
ρ	Ave/max imbalance	Ave distance	Ave/max imbalance	Ave distance
0.1	0.6149/5.004	1.7609	1.2116/7.002	1.5049
0.5	0.9924/7.004	1.5816	1.9970/9.004	1.5663
1	0.8387/7.004	1.4976	1.1275/8.004	1.5195
2	0.3369/6.004	1.3529	0.3611/6.004	1.3770

	MDP	*MDP*	*LVS*	*LVS*
ρ	Ave/max imbalance	Ave distance	Ave/max imbalance	Ave distance
0.1	0.6280/5.04	1.7608	1.2248/7.02	1.5049
0.5	1.0067/7.04	1.5681	2.0110/9.04	1.5663
1	0.8475/8.04	1.4891	1.1384/8.04	1.5195
2	0.3418/6.04	1.3540	0.3658/6.04	1.3770

	MDP	*MDP*	*LVS*	*LVS*
ρ	Ave/max imbalance	Ave distance	Ave/max imbalance	Ave distance
0.1	0.5836/5.4	1.5649	1.3560/7.2	1.5049
0.5	1.1460/8.4	1.5286	2.1510/9.4	1.5663
1	0.9572/8.4	1.4714	1.2475/8.4	1.5195
2	0.3878/6.4	1.3497	0.4126/6.4	1.3770

Table 12.6: The average and maximum imbalance, and maximum distance, recorded for a simulation run of 500,000 events (arrivals and departures) when $l = 8$, $c = 3$, and $\tau = 0.001, 0.01, 0.1$ (tables presented in order). The *MDP* policy and the last vacated/random space (*LVS*) allocation algorithms were used. The service time distribution was Erlang.

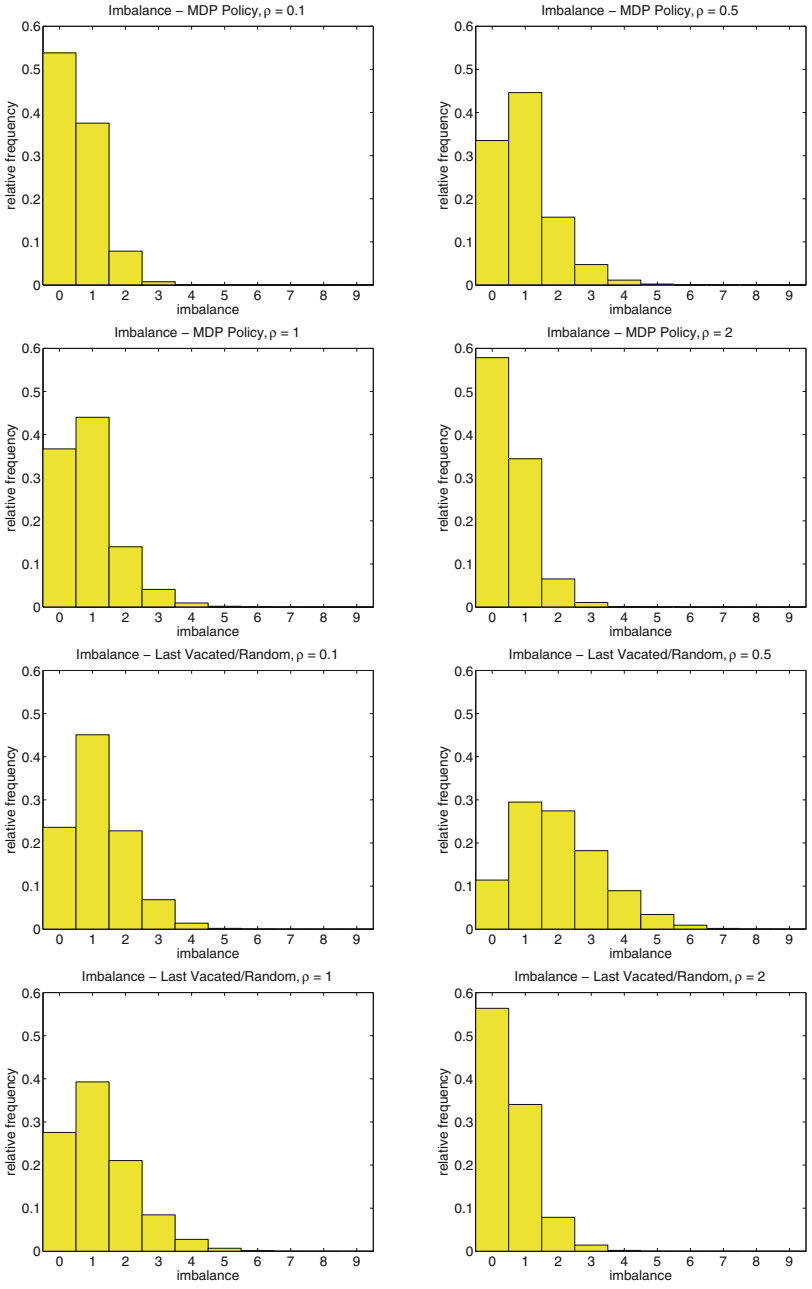


Fig. 12.7: Histograms of the imbalance measure when $l = 8$, $c = 3$, $\tau = 0.001$ and $\rho = 0.1, 0.5, 1, 2$. The service time distribution was exponential. The *top* four histograms are for the *MDP* Policy algorithm, and the *bottom* four are for the *LVS* heuristic algorithm

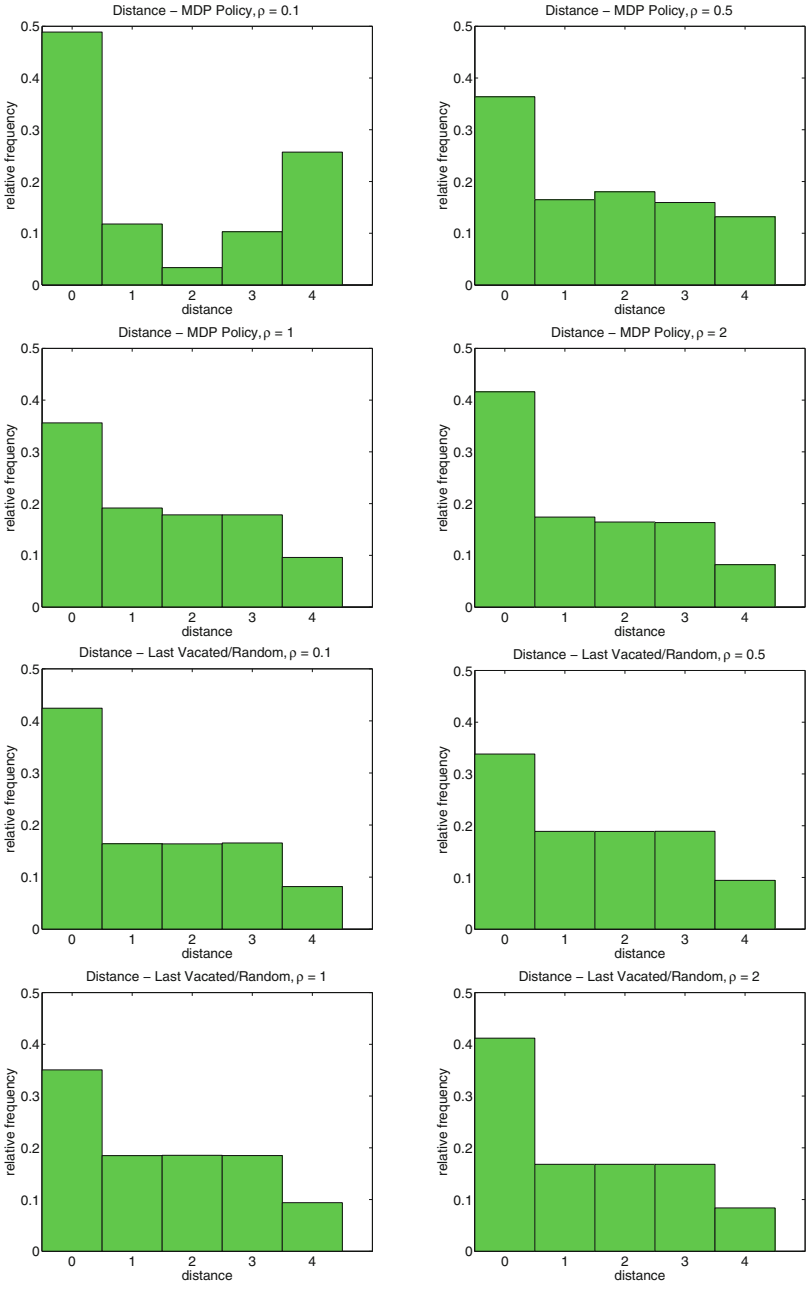


Fig. 12.8: Histograms of the distanced travelled when $l = 8$, $c = 3$, $\tau = 0.001$ and $\rho = 0.1, 0.5, 1, 2$. The service time distribution was exponential. The *top* four histograms are for the *MDP* policy algorithm, and the *bottom* four are for the *LVS* heuristic algorithm

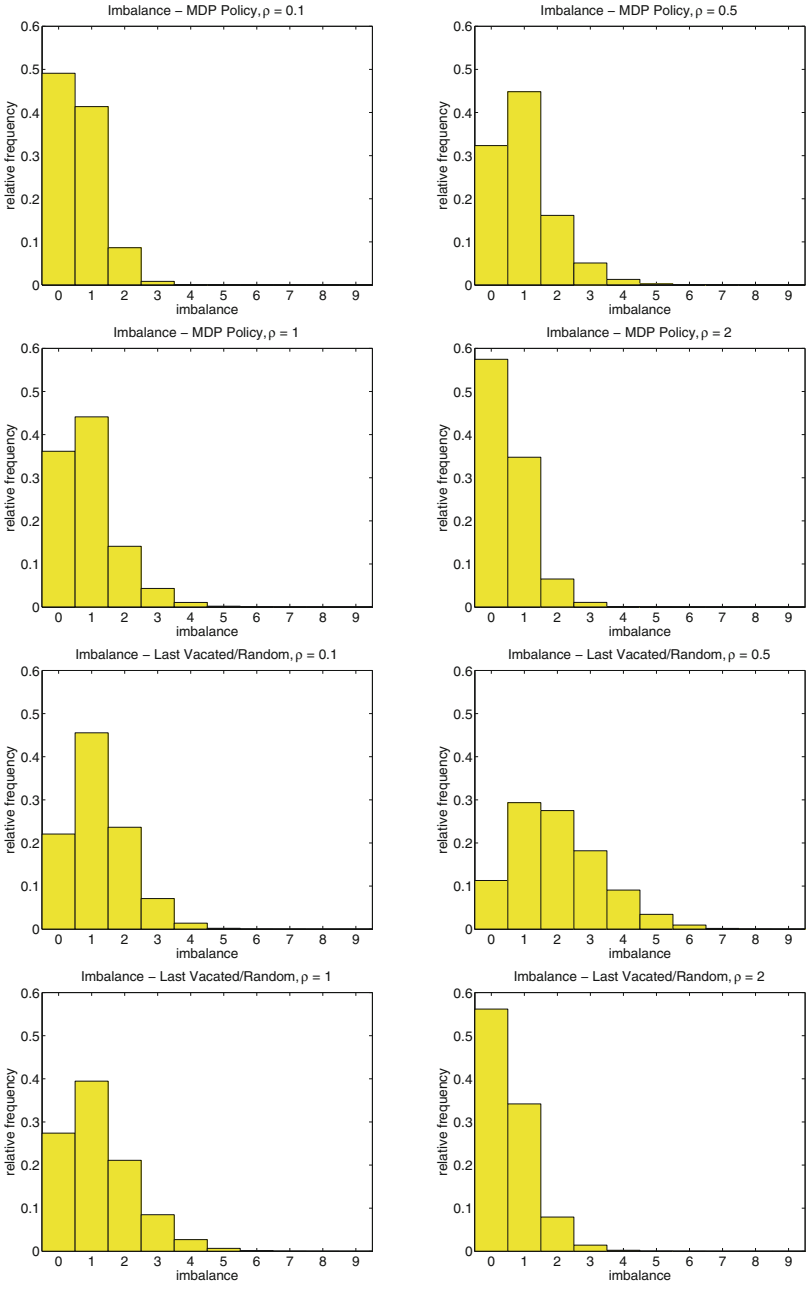


Fig. 12.9: Histograms of the imbalance measure when $l = 8$, $c = 3$, $\tau = 0.001$ and $\rho = 0.1, 0.5, 1, 2$. The service time distribution was Erlang. The *top* four histograms are for the *MDP* policy algorithm, and the *bottom* four are for the *LVS* heuristic algorithm

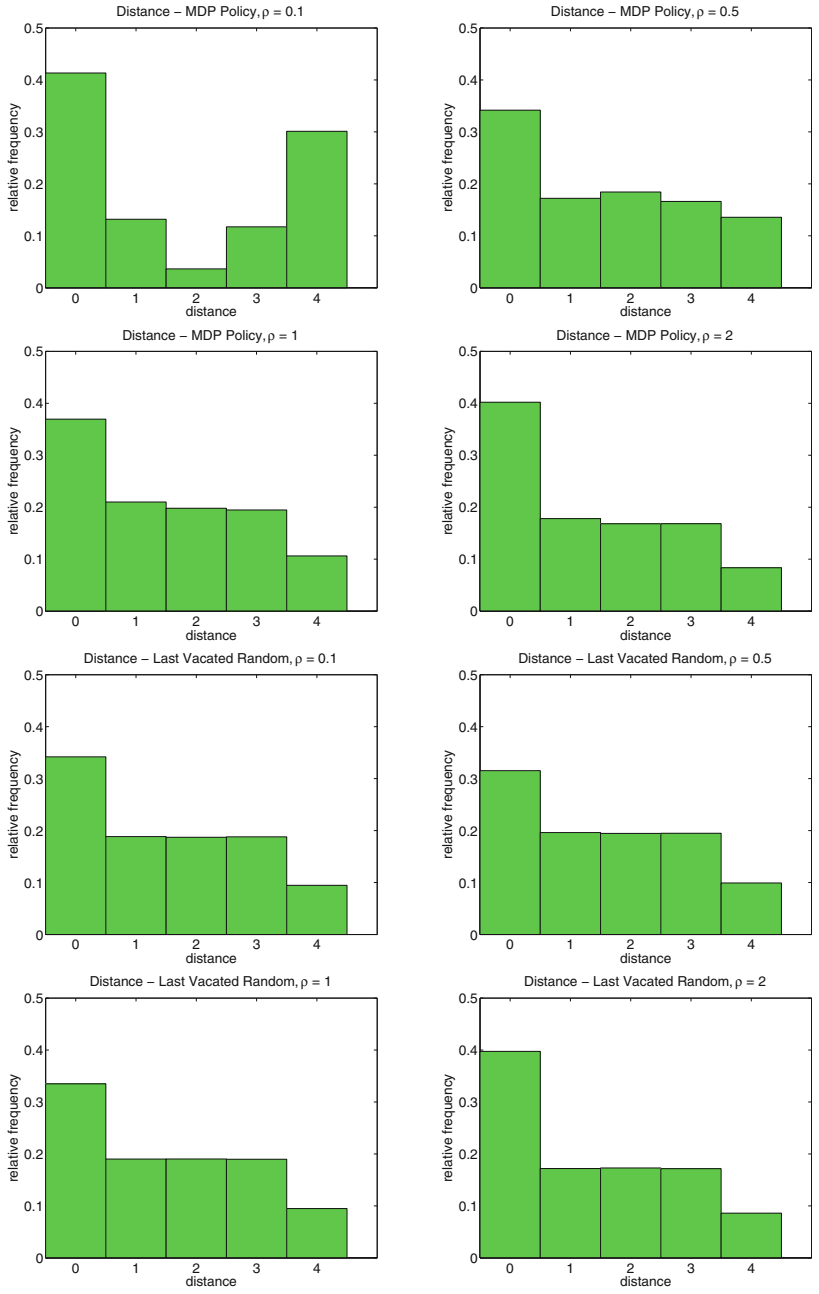


Fig. 12.10: Histograms of the distanced travelled when $l = 8$, $c = 3$, $\tau = 0.001$ and $\rho = 0.1, 0.5, 1, 2$. The service time distribution was Erlang. The *top* four histograms are for the *MDP* policy algorithm, and the *bottom* four are for the *LVS* heuristic algorithm

Simulation Results for $l = 20$ *and* $c = 1$

See Tables 12.7 and 12.8.

	MDP	*MDP*	*LVS*	*LVS*
ρ	Ave/max imbalance	Ave distance	Ave/max imbalance	Ave distance
0.1	0.6445/4.010	3.8097	1.3039/8.006	3.2123
0.5	1.1810/8.010	3.9467	2.2425/9.009	3.7836
1	0.9801/8.008	3.7175	1.2918/8.010	3.7197
2	0.3995/6.009	3.3250	0.4226/7.003	3.3612

Table 12.7: The average and maximum imbalance, and maximum distance, recorded for a simulation run of 500,000 events (arrivals and departures) when $l = 20$, $c = 1$, and $\tau = 0.001$. The *MDP* policy and the last vacated/random space (*LVS*) allocation algorithms were used. The service time distribution was exponential.

	MDP	*MDP*	*LVS*	*LVS*
ρ	Ave/max imbalance	Ave distance	Ave/max imbalance	Ave distance
0.1	0.6997/5.010	4.4734	1.3438/8.010	3.7009
0.5	1.2186/8.008	4.1198	2.2859/9.010	3.9410
1	1.0028/8.008	3.8305	1.2865/8.010	3.8184
2	0.3968/6.010	3.4077	0.4247/6.009	3.4611

Table 12.8: The average and maximum imbalance, and maximum distance, recorded for a simulation run of 500,000 events (arrivals and departures) when $l = 20$, $c = 1$, and $\tau = 0.001$. The *MDP* policy and the last vacated/random space (*LVS*) allocation algorithms were used. The service time distribution was Erlang.

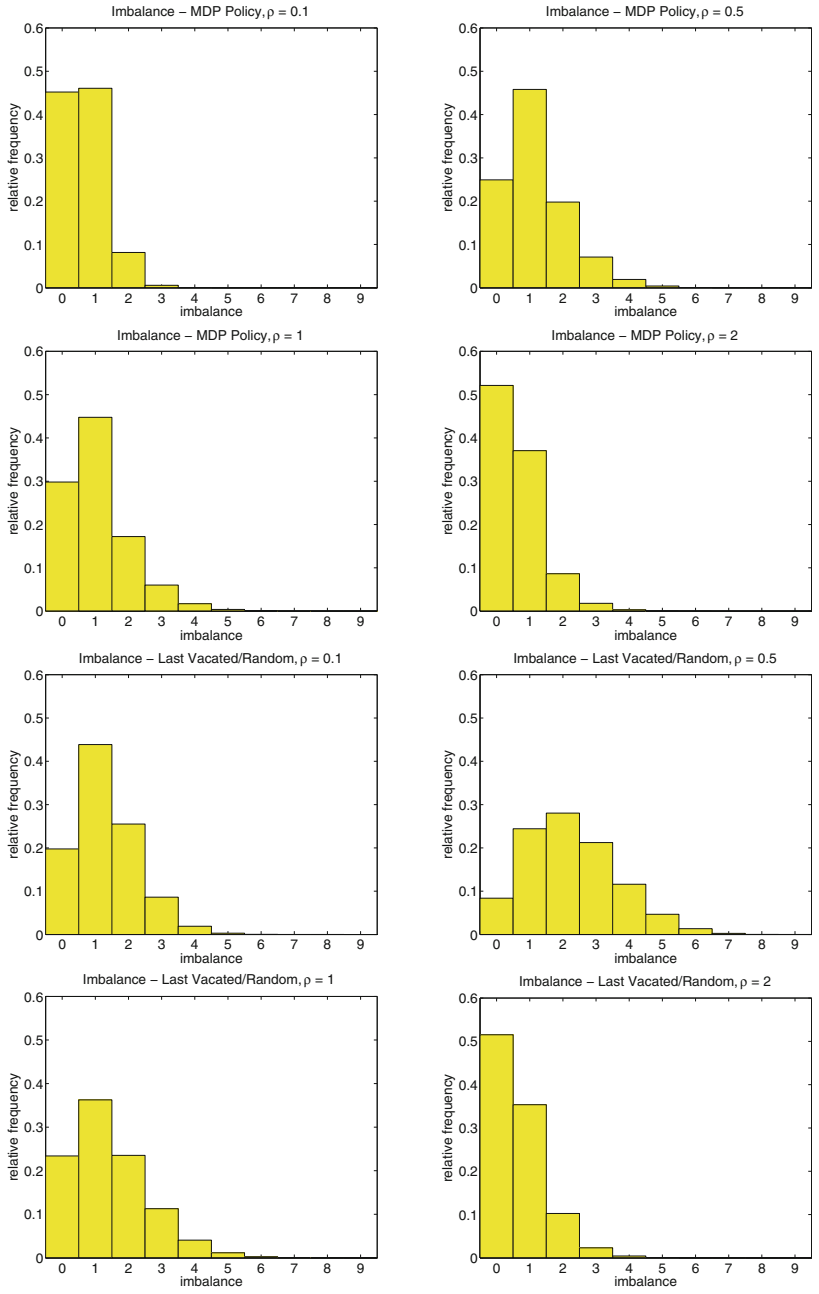


Fig. 12.11: Histograms of the imbalance measure when $l = 20$, $c = 1$, $\tau = 0.001$ and $\rho = 0.1, 0.5, 1, 2$. The service time distribution was exponential. The *top* four histograms are for the *MDP* Policy algorithm, and the *bottom* four are for the *LVS* heuristic algorithm

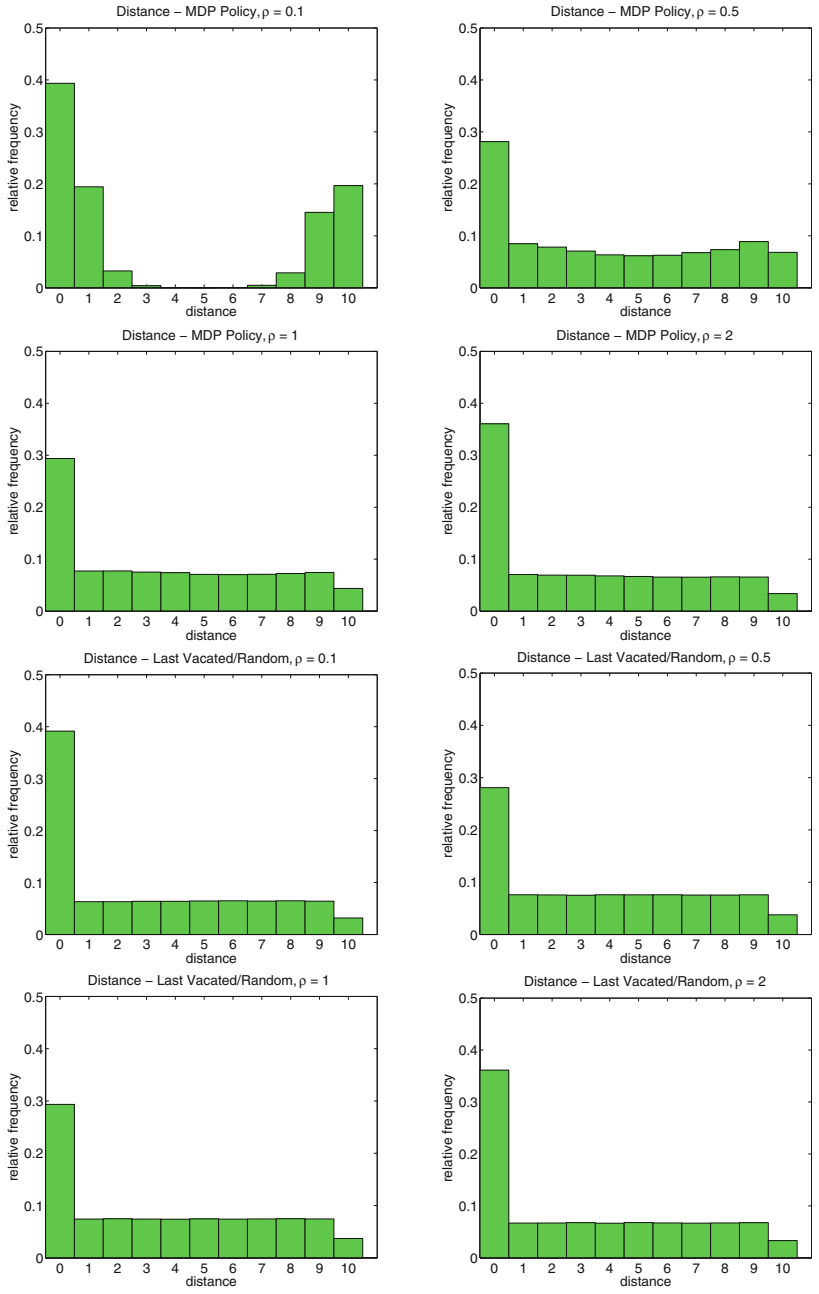


Fig. 12.12: Histograms of the distanced travelled when $l = 20$, $c = 1$, $\tau = 0.001$ and $\rho = 0.1, 0.5, 1, 2$. The service time distribution was exponential. The *top* four histograms are for the *MDP* policy algorithm, and the *bottom* four are for the *LVS* heuristic algorithm

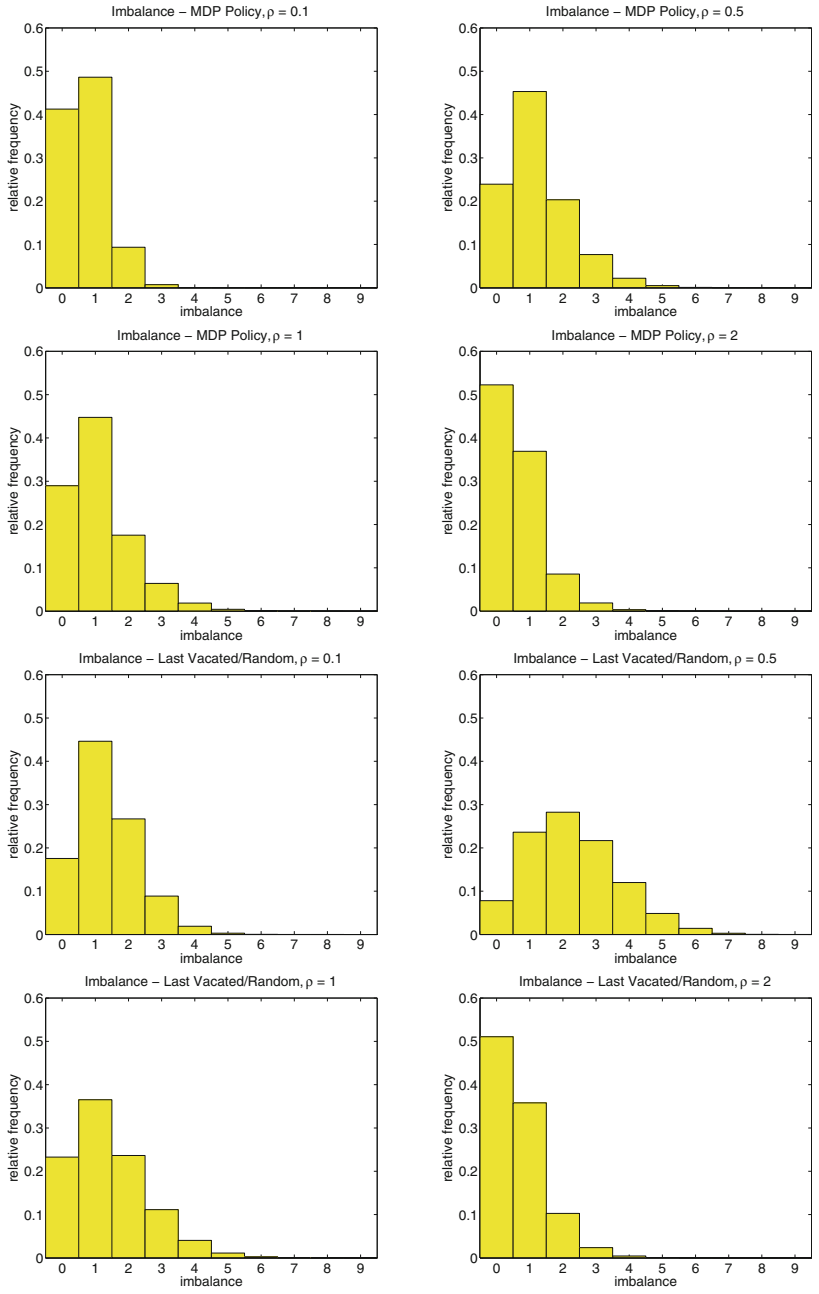


Fig. 12.13: Histograms of the imbalance measure when $l = 20$, $c = 1$, $\tau = 0.001$ and $\rho = 0.1, 0.5, 1, 2$. The service time distribution was Erlang. The *top* four histograms are for the *MDP* Policy algorithm, and the *bottom* four are for the *LVS* heuristic algorithm

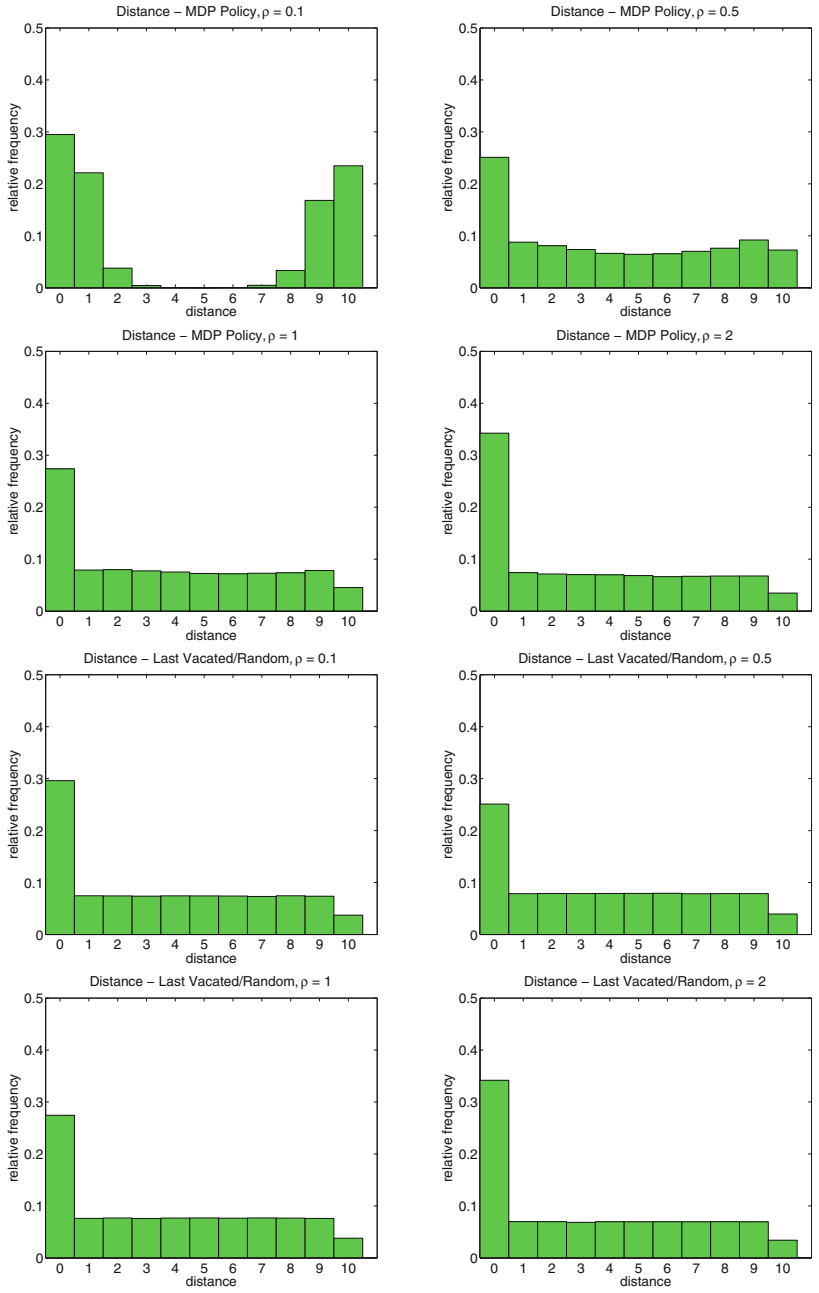


Fig. 12.14: Histograms of the distanced travelled when $l = 20$, $c = 1$, $\tau = 0.001$ and $\rho = 0.1, 0.5, 1, 2$. The service time distribution was Erlang. The *top* four histograms are for the *MDP* policy algorithm, and the *bottom* four are for the *LVS* heuristic algorithm

References

1. E. Altman, *Constrained Markov Decision Processes* (Chapman and Hall/CRC, Boca Raton, FL, 1999)
2. F. Beretta, Motor vehicle automatic parking system, and related improved silos structure Patent - US 5338145 A (1994). http://www.google.com.au/patents/US5338145. Accessed on 30 April 2015
3. H.S. Buch, Storage structure in particular a multi-story car park. Patent - US 5864995 A (1999). https://www.google.com/patents/US5864995. Accessed on 30 April 2015
4. L.F.A.M. Gomes, An operations research approach to the optimal control of parking systems. Found. Control Eng. **11**, 33–42 (1986)
5. H. Hwang, S. Lee, Expected service time model for a rotary parking system. Comput. Ind. Eng. **35**, 559–562 (1998)
6. D. Monahan, De-mystifying automated parking structures (2011). Reference 11 of Wikipedia, Automated parking system. http://en.wikipedia.org/wiki/Automated_parking_system. Accessed on 11 March 2015
7. A.R. Odoni, On finding the maximal gain for Markov decision processes. Oper. Res. **17**, 857–860 (1969)
8. W.B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality* (Wiley, Hoboken, NJ, 2011)
9. H. Schween, In a parking garage having more than one floor. Patent - US 5669753 A (1994). http://www.google.com/patents/US5669753. Accessed on 30 April 2015
10. H.A. Taha, *Operations Research: An Introduction* (MacMillan, New York, 1987)
11. D. Trevisani, Automated parking system and subassemblies therefor. Patent - US 5173027 A (1992). http://www.google.com/patents/US5173027. Accessed on 30 April 2015
12. L. Vita, Vehicle parking structure. Patent - US 5980185 A (1999). http://www.google.com.au/patents/US5980185. Accessed on 30 April 2015
13. Wikipedia, Automated Parking System. http://en.wikipedia.org/wiki/Automated_parking_system. Accessed on 10 March 2015
14. Wikipedia, Car Parking System. http://en.wikipedia.org/wiki/Car_Parking_System. Accessed on 10 March 2015
15. Wikipedia, Paternoster. http://en.wikipedia.org/wiki/Paternoster. Accessed on 10 March 2015
16. W.L. Winston, *Operations Research*, 4th edn. (Brooks/Cole: Belmont, CA, 2004)
17. YouTube, Revopark. Uploaded on 3 May 2007. url:https://www.youtube.com/watch?v=ApaiJ0xbmMA. Accessed on 10 March 2015
18. YouTube, Automated Rotary Parking System. Uploaded on 12 May 2009. url:https://www.youtube.com/watch?v=JXqzhmTi8Eo. Accessed on 10 March 2015

19. YouTube, Rotary Parking. Uploaded on 30 November 2010. url:https://www.youtube.com/watch?v=MsIHFyWuk4k. Accessed on 10 March 2015
20. YouTube, PTV Vissim: Automated Parking Simulation. Uploaded on 3 February 2011. https://www.youtube.com/watch?v=I1QSSWe8pV8. Accessed on 10 March 2015
21. YouTube, FATA Automated Parking Systems: 1 Car Park, 5 Systems, Uploaded on 2 August 2012. https://www.youtube.com/watch?v=VwS1QwXqgpk. Accessed on 10 March 2015

Chapter 13
Dynamic Control of Traffic Lights

Rene Haijema, Eligius M.T. Hendrix, and Jan van der Wal

Abstract Traffic lights are put in place to dynamically change priority between traffic participants. Commonly, the duration of green intervals and the grouping, and ordering in which traffic flows are served are pre-fixed. In this chapter, the problem of minimizing vehicle delay at isolated intersections is formulated as a Markov Decision Process (MDP). Solving the MDP is hampered by a large multi-dimensional state space that contains information on the traffic lights and on the queue lengths. For a single intersection, an approximate solution is provided that is based on policy iteration (PI) and decomposition of the state space. The approach starts with a Markov chain analysis of a pre-timed control policy, called Fixed Cycle (FC). The computation of relative states values for FC can be done fast, since, under FC, the multi-dimensional state space can be decomposed into sub-spaces per traffic flow. The policy obtained by executing a single iteration of Policy Iteration (PI) using relative values is called RV1. RV1 is compared for two intersections by simulation with FC, a few dynamic (vehicle actuated) policies, and an optimal MDP policy (if tractable). RV1, approximately solves the MDP, and compared to FC, it shows less delay of vehicles, shorter queues, and is robust to changes in traffic volumes. The approach shows very short computation times, which allows the application to networks of intersections, and the inclusion of estimated arrival times of vehicles approaching the intersection.

R. Haijema
Operations Research and Logistics group, Wageningen University, Wageningen, The Netherlands
e-mail: Rene.Haijema@wur.nl

E.M.T. Hendrix
Universidad de Málaga, Computer Architecture, Málaga, Spain
e-mail: eligius@uma.es

J. van der Wal
Faculty of Economics and Business, University of Amsterdam, Amsterdam, The Netherlands

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_13

13.1 Problem

Traffic lights are introduced to resolve conflicts between road users by dynamically changing the priority to cars approaching an intersection from different directions. In practice, the underlying optimization problem is not always clearly defined by policy makers. Several objectives are possible: minimize average delay, minimize pollution by cars (e.g., CO2 emission) or a combination. In addition, (political) constraints may apply such as public transport traveling at a separate lane has the highest priority over all other traffic flows. In practice, traffic engineers aim to set a good (hopefully nearly optimal) control scheme using simulation software, queueing delay formulas, and experience. For an overview of existing methods to control road traffic, see [9].

Many road users experience traffic lights as a source of delay. This chapter presents a model for the underlying Markov Decision Process (MDP) of minimizing the expected delay or waiting time of cars. The principle of the MDP model is to dynamically adjust the traffic lights depending on the number of cars queued in each queue, and the current state of the traffic lights. Accurate information on the number of queued cars is available to the controller from magnetic loop detectors cut into the surface of the road, or other sensors or cameras positioned along the road [1, 7, 12].

A number of dynamic control policies, reported in the literature, like SCOOT and SCAT [8], dynamically switch between off-line calculated fixed cycles (FC). The approach that we present in this chapter has more freedom to choose: it does not have to choose between pre-calculated time plans; instead it requires only one FC to be calculated off-line. Another class of dynamic control is vehicle actuated (VA) control policies, that are characterized by a minimum and a maximum green period for each traffic flow, and a gap-out time to dynamically decide to end a green period. The optimization of VA policies is complicated and usually requires heuristics, because of the number of parameters to be set jointly for all traffic flow (see [14]).

Also solving an MDP for infrastructures with many traffic flows, requires heuristics solution procedures. Because of the curse of dimensionality, the number of states grows exponentially in the number of queues. Parallelization of algorithms to solve MDPs, as in [6], provides (at best) a linear reduction of the solution time, which is not enough to solve many complex infrastructures in practice. Two general techniques to solve large scale MDPs are aggregation and decomposition to reduce the state space. Decomposition has been applied successfully to other problem settings, see [13] and [2]. Another method is named Approximate Dynamic Programming, which is based on approximating the value function [10]. In this chapter, we present an approach based on decomposition.

Section 13.2 describes the MDP model. Section 13.3 describes a way to approximately solve the MDP. The resulting policy is evaluated by simulation in Sect. 13.4. A short discussion and some conclusions follow in Sect. 13.5. In Appendix of this chapter follows a summary of notations.

13.2 Markov Decision Process (MDP)

The model includes the flows of cars at an intersection, excluding other participants in traffic, such as pedestrians and public transport. The applied definition of waiting time is the time a car spends in the queue regardless of the color of the light. The corresponding optimization is modeled in terms of an MDP. Approximate solutions of the model are derived via policy iteration in Sect. 13.3. The resulting traffic control tables are then simulated in Sect. 13.4.

13.2.1 Examples: Terminology and Notations

To get familiar with the notation, consider the infrastructures in Fig. 13.1. Figure 13.1a depicts a simple intersection with only $F = 4$ traffic flows leading to four queues. The flows and queues are numbered clockwise: 1–4. Flows 1 and 3 are grouped into combination 1 (C_1), as they receive green simultaneously. Combination 2 (C_2) consists of flows 2 and 4. At most one combination at a time has right of way (when its lights are green or yellow). When switching from green to one combination to green to the other combination, the green lights are first changed into yellow (for two time slots) followed by a clearance period (of one time slot) during which all lights are red and after which lights of some combination are changed into green. The clearance time is included to safely clear the intersection. The recurring questions are when to end a green period, and which combination

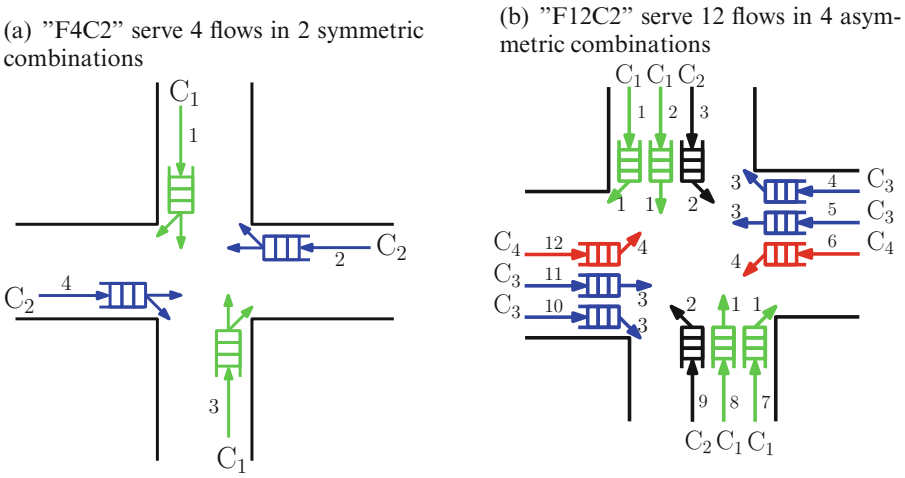


Fig. 13.1: Two typical infrastructures. (**a**) "F4C2" serve four flows in two symmetric combinations. (**b**) "F12C2" serve 12 flows in four asymmetric combinations

to serve next, given the actual color of the traffic lights, the actual number of cars present at each queue, and the probabilistic arrival process of new cars.

Figure 13.1b shows a more complex intersection where $F = 12$ flows are grouped into four combinations C_1–C_4. C_1 and C_3 consist of twice as many flows than C_2 and C_4. As long as some cars are waiting at all queues, giving priority to C_1 or C_3 results in more departures per unit time than giving priority to C_2 and C_4. The decisions *'when-to-end-a-green-period'* as well as *'which-combination-to-serve-next'*, should depend on the number of cars waiting at each queue: $\mathbf{q} = (q_1, q_2, \ldots, q_{12})$. An optimal policy prescribes for each possible state of the queues, i.e. each value of the vector $\mathbf{q}$, the action to take given the current state of the lights l. Such an optimal policy can be obtained from the solution of the underlying MDP.

13.2.2 MDP Model

To model the decision problem, time is modeled in time slots of a fixed length equal to a safe traveling distance between two cars. Hence we set the length of a time slot to 2 s, which correspond roughly with the time between two departures from a queue.

13.2.2.1 State

The state (at the start of a slot) is $\mathbf{s} = (l, \mathbf{q})$, where $l \in L$ is the state of the traffic lights. S denotes the state space, which is composed of the state space of the traffic lights L and the state space related to the queue lengths. L is a finite set of $|L|$ elements. The number of cars queued is in theory unbounded. However, for computational reasons, we limit the length of each queue to $Q-1$ cars. The number of queue states is Q^F. The total number of states is $|S| = |L|Q^F$, which grows exponentially in the number of traffic flows F.

13.2.2.2 Action

The action $a \in A(\mathbf{s}) \subseteq L$, taken at the start of a slot, changes the traffic light state instantaneously, i.e. immediately after observing the state s. As switching between green for some (combination of) flow(s) to green to another, takes time to clear the intersection the choices for adjusting the lights is limited to $A(\mathbf{s}) \subseteq L$.

13.2.2.3 State Transition Probabilities

The state changes by the action choice on the traffic light, and by having cars departing or arriving at the queues. From each queue with one or more cars waiting, and

that has priority according to light state l, exactly one car will leave within a time slot. Within a time slot, new cars arrive at the queues by F independent Bernoulli processes: i.e. with probability λ_f, a new car arrives at flow f (and with probability $1-\lambda_f$ no car arrives). The car either forms or joins a queue, or, if having priority and no car is queued on his lane, it crosses the intersection in the same slot without delay.

The state transition probability from state $\mathbf{s}$ to state $\mathbf{s}' = T(\mathbf{s},a) = (a,\mathbf{q}')$ taking action a is

$$P(\mathbf{q}'|\mathbf{s},a) = \prod_{f=1}^{F} p_f(q'_f|(l,q_f),a) \tag{13.1}$$

where $p_f(q'_f|(l,q_f),a)$ depends on l and whether a car arrives (w.p. λ_f) or not, as follows:

- when l implies priority (green or yellow) to flow f:

$$p_f(q'_f|(l,q_f),a) = \begin{cases} \lambda_f & \text{if } q'_f = q_f, \\ 1-\lambda_f & \text{else if } q'_f = \max\{0,q_f-1\} \\ 0 & \text{otherwise.} \end{cases} \tag{13.2}$$

- when l implies no priority (=red) to flow f:

$$p_f(q'_f|(l,q_f),a) = \begin{cases} \lambda_f & \text{if } q'_f = q_f+1 \\ 1-\lambda_f & \text{else if } q'_f = q_f \\ 0 & \text{otherwise.} \end{cases} \tag{13.3}$$

13.2.2.4 Contribution: Waiting Costs

The objective is to minimize the expected number of cars queued at the start of a time slot, such that by Little's law the expected waiting time is minimized. The contribution in a time slot to this objective function is the so-called one-period or direct cost function:

$$c(\mathbf{q}) = \sum_{f=1}^{F} q_f. \tag{13.4}$$

13.2.2.5 Bellman Equation

A stationary dynamic control policy π^* that minimizes the expected number of queued cars fulfils

$$\pi^*(\mathbf{s}) = \arg\min_{a\in A(\mathbf{s})} \sum_{\mathbf{q}'} P(\mathbf{q}'|\mathbf{s},a))v^*(\mathbf{s}'). \tag{13.5}$$

where there exists a constant g^* and a value function $\mathbf{v}^*$ being a solution of the Bellman equation:

$$\forall \mathbf{s} \in S: \quad v^*(\mathbf{s}) + g^* = c(\mathbf{q}) + \min_{a \in A(\mathbf{s})} \sum_{\mathbf{q}'} P(\mathbf{q}'|\mathbf{s}, a)) v^*(\mathbf{s}') \tag{13.6}$$

The constant g^* represents the expected number of cars waiting at the start of a time slot when an optimal policy π^* is followed.

At this point, we have to make a technical remark: to get a finite state space S, we have limited each queue length to at most $Q-1$ cars. However, in the Bellman equation we have included transitions to states $\mathbf{s}'$ that are beyond the state space. (e.g. when a car is arriving at a queue that contains already $Q-1$ cars). To determine $v(\mathbf{s}')$ for these states we apply quadratic extrapolation. For details see [3].

13.2.2.6 Computational Complexity

Solving the Bellman equation involves solving a set of $|S|$ equations in $|S|+1$ unknowns; therefore one has one degree of freedom to fix one of the elements of $\mathbf{v}^*$. Hence $v^*(\mathbf{s})$ is a relative value of state $\mathbf{s}$ when an optimal policy π^* is applied. The Bellman equation can be solved using fixed point algorithms, such as value iteration, or by exact algebraic methods. A detailed discussion of the theory and methods to solve MDPs is found in [11].

However, solving the MDP is only possible for infrastructures with a 'small' number of traffic flows, and under non-saturated conditions such that a reasonable bound to the queue lengths can be set. For example, for infrastructure F12C4, when all 12 queues may have up to 9 cars, then each element q_f can take 10 possible values (0–9). Consequently, the number of possible vectors $\mathbf{q}$ is 10^{12}. Hence the state space of the MDP easily exceeds the computationally acceptable limit of say 10 million states. The computational limit can be extended a bit by parallelization [6]. However, that only partly solves the computational issue, as the number of states grows exponentially with the number of traffic flows F. Large MDPs that cannot be solved to optimality require an approximate solution method, like the one that we discuss in the next Section.

13.3 Approximation by Policy Iteration

13.3.1 Policy Iteration (PI)

Instead of applying exact algebraic methods to solve Eq. (13.6), we apply a policy iteration (PI) algorithm to approximate an optimal policy. PI successively repeats the following two steps.

Step 1 *Policy evaluation step:* for an initial policy π, determine for all $\mathbf{s} \in S$ the associated relative state values $v^{\pi}(\mathbf{s})$ that satisfy:

$$v^{\pi}(\mathbf{s}) + g^{\pi} = c(\mathbf{q}) + \sum_{\mathbf{q}'} P(\mathbf{q}'|\mathbf{s}, \pi(\mathbf{s})))v^{\pi}(\mathbf{s}'). \tag{13.7}$$

Step 2 *Policy improvement step:* Next, policy π is improved by executing a policy improvement step:

$$\pi'(\mathbf{s}) = \arg\min_{a \in A(\mathbf{s})} \sum_{\mathbf{q}'} P(\mathbf{q}'|\mathbf{s}, a))v^{\pi}(\mathbf{s}'). \tag{13.8}$$

if $\pi' = \pi$, then stop (as $\pi = \pi^*$), otherwise set $\pi := \pi'$ and return to Step 1.

Just as any other methods to solve the Bellman equations, PI suffers from the computational burden due to the large state space. The advantage of PI is that doing only one iteration may already give a good approximation when the initial policy is reasonably good.

13.3.2 Initial Policy: Fixed Cycle (FC)

A well studied policy is a static policy for which one pre-fixes the time intervals at which flows get green and the cyclic order in which combinations of flows are served. The cycle has a fixed length of D time units, which is the sum of the green periods and the time period needed to switch between combinations. Such a policy we call FC. The slots within a cycle are numbered $t = 1, 2, \ldots, D$. The slot number provides all relevant information about the state of the traffic lights: i.e. from t one can derive for each flow f the color of the light, the time it takes till getting green, yellow, or red. Thus the state of the traffic light l is for $\pi = \text{FC}$ given by t.

FC could act as an initial policy for PI. Therefore one first needs to configure FC: i.e. set D, (nearly) optimal lengths of the green periods, and the cyclic order in which combinations get priority. An optimization algorithm for setting an initial FC is presented in [3] and [4].

13.3.3 Policy Evaluation Step of FC

The relative state values of FC, $v^{FC}(\mathbf{s})$, can be decomposed in relative state values $v_f^{FC}(t, q_f)$ per traffic flow f:

$$v^{FC}(\mathbf{s}) = \sum_{f=1}^{F} v_f^{FC}(t, q_f). \tag{13.9}$$

Relative state values $v_f^{FC}(t,q_f)$ are determined by value iteration:

Step 1a. Define $V_0^f(t,q)=0$ for all $t\in\{1,\ldots,D\}$ and $q\in\{0,\ldots,Q\}$ and let ε take a very small value (compared to the expected number of cars waiting at the start of any time slot), e.g. 10^{-5}.

Step 1b. For all $t\in\{1,\ldots,D\}$ and $q\in\{0,\ldots,Q\}$, recursively compute $V_{n+1}^f(t,q)$ as follows (with $V_n^f(D+1,\cdot)$ read as $V_n^f(1,\cdot)$):
Start with $n=0$ and $\mathbf{V}_0=\mathbf{0}$.
Repeat

- if flow f is having priority (green or yellow) during time slot t:

$$V_{n+1}^f(t,q):=q+\lambda_f\cdot V_n^f(t+1,q)+(1-\lambda_f)\cdot V_n^f(t+1,(q-1)^+), \tag{13.10}$$

  where $x^+=\max\{0,x\}$,
- if flow f is not having priority during time slot t (its light is red):

$$V_{n+1}^f(t,q):=q+\lambda_f\cdot V_n^f(t+1,q+1)+(1-\lambda_f)\cdot V_n^f(t+1,q)\,. \tag{13.11}$$

- $n:=n+1$;

until $\max(\mathbf{V}_n^f-\mathbf{V}_{n-D}^f)-\min(\mathbf{V}_n^f-\mathbf{V}_{n-D}^f)<\varepsilon$.
Set $N:=n$.
For all $t\in\{1,\ldots,D\}$ and $q\in\{0,\ldots,Q\}$, the difference $(V_N^f(t,q)-V_{N-D}^f(t,q))$ is at most ε off from the long-run average cost per cycle ($g^{(f)}$), as the D-step Markov chains are all irreducible.

Step 1c. We set the relative state values relative to a fixed reference state, $(D,0)$. Thus $\mathbf{v}_f^{FC}$ is:

$$\mathbf{v}_f^{FC}\equiv\frac{\mathbf{V}_{N-D+1}^f+\cdots+\mathbf{V}_N^f}{D}-\frac{V_{N-D+1}^f(D,0)+\cdots+V_N^f(D,0)}{D}\cdot\mathbf{1}\,, \tag{13.12}$$

where $\mathbf{1}$ is the all-ones vector.
A discussion of the relative value definition in (13.12) is found in [3]. The differences $V_N^f(q,t)-V_N^f(D,0)$ cannot be used for this purpose as FC is a periodic policy and consequently these differences change periodically with N.
An alternative criterion is to compare $\sum_{d=0}^{D-1}\mathbf{V}_{N-d}^f/D$ against the average cost over N slots:

$$\mathbf{v}_f^{FC}\equiv\frac{\mathbf{V}_{N-D+1}^f+\cdots+\mathbf{V}_N^f}{D}-N\cdot g\,, \tag{13.13}$$

where g may be approximated by any element of $\frac{\mathbf{V}_N^f-\mathbf{V}_{N-D+1}^f}{D}$, as N is sufficiently large.

For example, Fig. 13.2 shows for a particular queue state $\mathbf{q} = (4,2,2,1)$ at infrastructure F4C2, the valuation of the traffic light state t. Figure 13.2a shows for each of the four flows, the individual preference of time slot t. Figure 13.2b shows the sum of these preferences. On top of the x-axis (which shows the time slots of FC), labels are added that indicate which of the two combination gets green (G1/G2), or yellow (Y1/Y2), or whether all lights are red (R).In this case with $\mathbf{q} = (4,2,2,1)$, the best time slot is slot 1 (i.e. first slot of green to combination 1), as it gives the lowest relative (cost) value. FC loops over all time slots: after slot 12 (R=all red) it continues at slot 1 (G1=green to combination 1).

13.3.4 Single Policy Improvement Step: RV1 Policy

The relative value $v_f^{FC}(\tau, q_f)$ quantifies the preference of flow f for time slot τ, when q_f cars are waiting at queue f. Assuming all flows are equally important, the overall relative appreciation of time slot τ is the sum $\sum_{f=1}^{F} v_f^{FC}(\tau, q_f)$, as depicted in Fig. 13.2b.

By applying a single policy improvement step, one finds a new policy π' that may interrupt or breaks FC by dynamically deciding to decrease or increase a green period. That is, in state $(t, \mathbf{q})$, policy π' switches the state of the traffic light to the best time slot reachable from the current slot t:

$$\pi'(t, \mathbf{q}) = \arg \min_{\tau \in A(t,\mathbf{q})} \sum_{f=1}^{F} v_f^{FC}(\tau, q_f) \,, \tag{13.14}$$

where $A(t, \mathbf{q})$ is the set of time slots to which one may switch safely from the current traffic light state t.

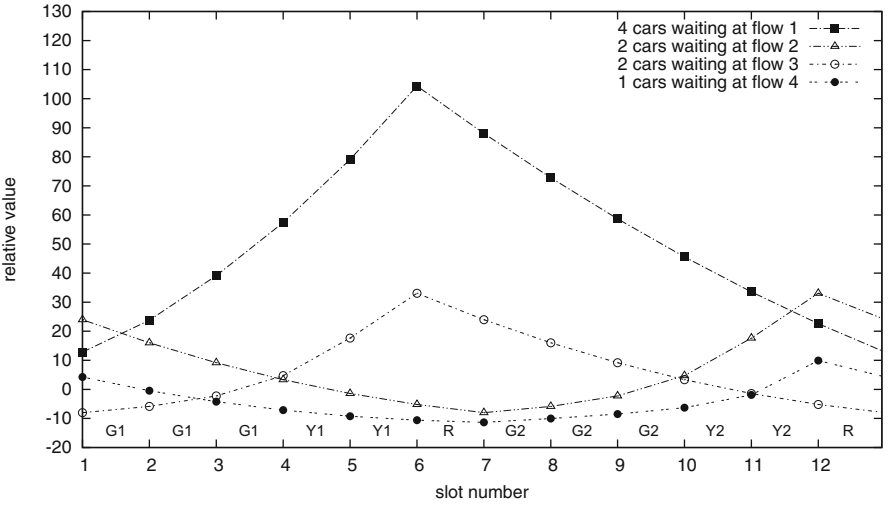
This process is illustrated by Fig. 13.2b: one selects the time slot that can be reached from the current time slot t, and that yields the lowest sum of relative values. That is, under cyclic control, if $t = 1$, 2, or 3, then $\pi'(t, \mathbf{q}) = 1$. If $t = 9$ or 10, then $\pi'(t, \mathbf{q}) = 10$. For $t = 8$ holds $\pi'(t, \mathbf{q}) = 9$, as lights cannot switch from red to yellow without granting first green to one combination. For all other values of t holds $\pi'(t, \mathbf{q}) = t$, i.e., one may not interrupt FC during switching.

We call this policy in the rest of this chapter the RV1 policy as it follows from a 1-step policy improvement using the relative values of FC.

13.3.5 Computational Complexity of RV1

The computational complexity of determining RV1 is very low as the state space under FC is effectively decomposed into the state per traffic flow. The computation of the relative values for each flow can be done quickly off-line. It allows a high

(a) Relative value curves for flows 1 to 4 when $\mathbf{q} = (4,2,2,1)$ cars are waiting.



(b) Sum of the relative value curves when $\mathbf{q} = (4,2,2,1)$ cars are waiting at flow 1 to 4.

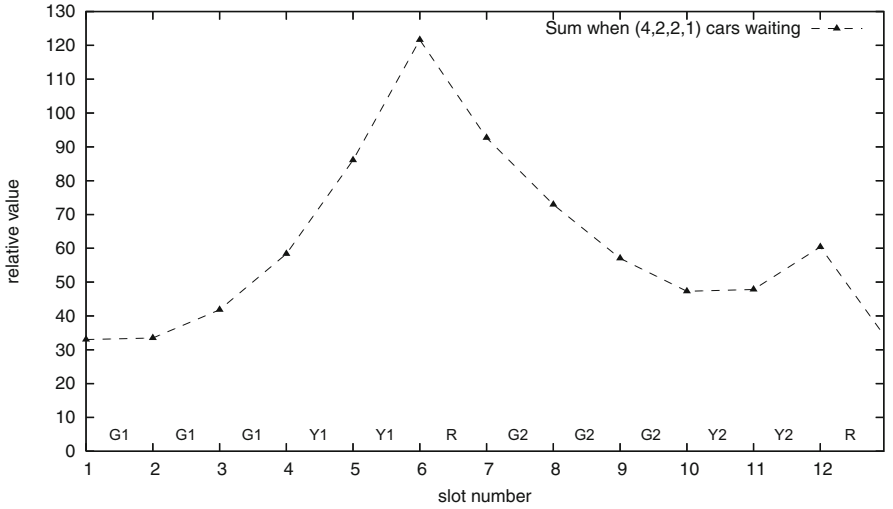


Fig. 13.2: Relative state values of FC. (**a**) Relative value curves for flows 1–4 when $\mathbf{q} = (4,2,2,1)$ cars are waiting. (**b**) Sum of the relative value curves when $\mathbf{q} = (4,2,2,1)$ cars are waiting at flow 1–4

bound on the queue length that is practically not restrictive. The computation of the sum of relative value curves is to be done quickly online: based on the actual number of cars queued at each queue and the actual state of the traffic lights, the best time slot to jump to can be computed in real time using Eq. (13.14). This is an important characteristic that allows applying RV1 to large and complex infrastructures.

13.3.6 Additional Iterations of PI

For problems with a large state space S, additional iterations of PI are not considered, since this is not possible to determine relative state values for policy RV1: i.e. $v^{RV1}(\mathbf{s})$ cannot be computed in reasonable time, as its state space cannot be decomposed. For problems with a small state spaces that allows additional iterations of PI (without decomposition of the state space), one may continue PI by following the procedure sketched in Sect. 13.3.1 until an optimal MDP policy is found.

13.4 Results

To assess the quality of RV1, its performance is evaluated by simulation and compared against FC and some exhaustive control policies. Exhaustive control (XC) grants green to a combination as long as cars are queued. Policy XC-1 switches to yellow as soon as at each 'green' queue at most one car is waiting anticipating its departure during the next (=first) yellow slot. XC-2 switches to yellow as soon as at each 'green' queue (at most) two cars are waiting, and thus XC-2 anticipates their departure during the next two yellow slots.

For both infrastructures F4C2 and F12C4 we consider symmetric cases: i.e. the arrival rates are identical for all flows. For F4C2 an optimal MDP policy is evaluated. For F12C4, the optimal MDP policy could not be determined due to the large state space. Acyclic control, non-identical arrival rates, and other infrastructures are reported in [3] and [5].

13.4.1 Simulation

The simulation model relies on the same assumptions as the MDP model. The accuracy of the results presented in the next subsections is primarily set by the number of simulation runs and the length of each run. All reported simulation results are based on 100 runs of 72,000 slots per run, corresponding to 4000 h in the real system. At the start of each run, a warming-up period of 450 slots (= 15 min) is applied. The reported mean waiting times are accurate up to (at least) 2–3 digits. To be concise, we do not report confidence intervals.

13.4.2 Intersection F4C2

Table 13.1 shows the results for the fully-symmetric F4C2 intersection at varying workloads ρ. In the cases of a workload of $\rho = 0.4, 0.6, 0.8$ all flows have identical arrival probabilities of respectively $\lambda_f = 0.2, 0.3, 0.4$. The workload ρ should be well below 1, to accommodate time for switching between combinations.

The Optimize-fixed-cycle algorithm (see [3, 4]) suggests cycle lengths of 8, 12 and 22 slots respectively, as reported in the next-to-last row (in seconds). The effective green time of a combination is the length of the related green and yellow period under FC. RV1 is based on FC with these cycle lengths and effective green times.

Table 13.1: Overall mean waiting time (in s) for the fully-symmetric F4C2

Rule	$\rho = 0.4$	$\rho = 0.6$	$\rho = 0.8$
RV1	5.06	7.01	14.2
FC	5.43 +7%	8.27 +18%	17.0 +20%
XC	5.76 +14%	8.82 +26%	19.9 +40%
XC-1	5.03 −1%	7.21 +3%	15.5 +9%
XC-2	5.09 +1%	7.31 +4%	14.2 +0%
MDP cyclic	4.89 −3%	6.95 −1%	13.5 −5%
FC cycle length (in s) and effective green times per combination	16 (6, 6)	24 (10, 10)	44 (20, 20)

The performance of the cyclic RV1 strategy obtained by the one-step policy improvement algorithm, is close to that of the optimal cyclic MDP strategy. Next to the average waiting times, we report the relative difference compared to RV1. The cyclic MDP policy performs only slightly better. FC and XC yield on average 15 and 27% larger waiting times; when the load is high ($\rho = 0.8$) the biggest differences can be observed. On average, RV1 is just slightly better than the anticipating exhaustive variants (XC-1 and XC-2). However, the differences are small for this simple fully-symmetric case.

13.4.3 Complex Intersection F12C4

For infrastructure F12C4, the optimal MDP strategy cannot be computed, because the number of states is prohibitively large. Even if the MDP model would truncate queues at the unrealistic level of two cars, the total number of states is still quite large: $8.5 \cdot 10^6$ states (= $(1+2)^{12}$ queue states times 16 traffic light states). F12C4 is not only computationally more complex because of the number of states, but also

because the combinations are asymmetric: C_1 and C_3 consist of four flows, whereas C_2 and C_4 consists of two flows only. It seems to be more profitable to under-serve combinations C_2 and C_4, since serving the other two combination results in more departures per time slot as long as cars are present at the respective queues.

The asymmetry in the number of flows per combination, makes it more difficult to define simple rules that perform well. We keep the same definition of exhaustive control: all queues must be empty before the green signals are turned into yellow. Under XC-2, green lights are turned into yellow as soon as at each of the flows that have right of way less than three cars are queued.

In Table 13.2 the results for varying workloads at a F12C4 intersection are presented for the case where all flows have identical arrival intensities $\lambda_f = 0.1$, 0.15, and 0.2, providing a workload of $\rho = 0.4$, 0.6 and 0.8 respectively. RV1 outperforms all other strategies. When the workload of the intersection is low (0.4) XC-2 performs equally well. At a high load XC-2 is too simplistic. At $\rho = 0.8$, FC performs even better than XC-2: XC-2 yields an average waiting time that is 28% higher than under RV1.

Table 13.2: Overall mean waiting time (in s) at different loads for F12C4 ($\lambda_f = \rho/4$)

Rule	$\rho = 0.4$		$\rho = 0.6$		$\rho = 0.8$	
RV1	13.5		19.3		41.8	
FC	15.0	+11%	23.7	+23%	50.5	+21%
XC	19.2	+42%	33.4	+73%	89.8	+115%
XC-1	14.9	+10%	25.1	+30%	70.1	+68%
XC-2	13.5	+0%	19.6	+2%	53.3	+28%
FC cycle length (in s)	32		40		88	
FC effective green times (in sec.)	(6, 6, 6, 6)		(8, 8, 8, 8)		(20, 20, 20, 20)	

Equal Allocation of Waiting Time

In Table 13.3, we study the waiting time at the different flows under a heavy workload of $\rho = 0.8$. Although the arrival rates λ_f are identical for all flows, the mean waiting time differs per combination. Combinations C_1 and C_3 are 'thicker' than combinations C_2 and C_4, since the latter two combinations have only two flows each, whereas C_1 and C_3 consists of four flows each. Therefore C_1 and C_3 experience a lower waiting time than combinations C_2 and C_4 under all policies except FC. The average waiting time per car under FC is identical for each flow as each flow experiences an effective green time of 20 s (10 slots) per cycle.

Table 13.3: Mean waiting times (in s) for symmetric F12C4 at $\rho = 0.8$

Rule	*EW* overall		*EW* C_1, C_3	*EW* C_2, C_4
RV1	41.8		37.4	50.6
FC dep. times 20, 20, 20, 20 s	50.5	+21%	50.5	50.4
XC	89.8	+115%	88.5	92.4
XC-1	70.1	+68%	68.9	72.4
XC-2	53.3	+28%	52.1	55.8

Although FC seems to be most fair in the sense that the flows experience similar average waiting times, RV1 performs much better: the average waiting time to flows of C_1 and C_3 is 13 s (or 26%) lower than under FC, while the average waiting times to C_2 and C_4 are virtually the same as under FC. Notice further that both XC and anticipative-exhaustive control (XC-1 and XC-2) perform worse than FC, when the workload is high.

13.5 Discussion and Conclusions

The dynamic control of traffic lights can be formulated as an MDP. In practice, for many infrastructures the state space may be too large to determine an optimal MDP policy. Nevertheless, this chapter shows that the principles and theory of MDP can still be applied to obtain good approximate solutions by executing a single iteration of a policy improvement (PI) algorithm. Key to this approach is to start PI with a well structured policy for which relative values of the state can be determined. For the problem of controlling traffic lights, such a well-structured policy is fixed cycle control (FC). Under FC the computation of relative values can be decomposed in computing relative values of the traffic light state.

For a single intersection, an approximate solution is provided that is based on policy iteration (PI) and decomposition of the state space. The approach starts with a Markov chain analysis of a pre-timed control policy, called Fixed Cycle (FC). The computation of relative states values for FC is fast as under FC the multi-dimensional state space can be decomposed into sub-spaces per traffic flow. The policy obtained by executing a single iteration of PI using relative values of FC, is called RV1.

Numerical results obtained by simulation, shows RV1 greatly reduces the average waiting time compared to FC (and other policies). As the relative values of states under policy RV1 cannot be computed using decomposition, additional PI steps can be executed only for infrastructures for which the state space is not too large.

RV1 seems to be a promising policy for practical application as:

- RV1 is robust to changes in traffic volumes: RV1 performs well even when the underlying FC is sub-optimal,
- RV1 is thus easy to maintain: if traffic conditions change the performance of RV1 deteriorates less rapidly than FC,
- RV1 is fast: evaluating a change of the traffic lights is done online in a micro second; also the off-line calculation of the relative values for each flow can be done quickly,
- RV1 can be extended to include information on the predicted arrival time of a car approaching the intersection,
- RV1 can be scaled up to complex intersection and networks of intersections.

Future research may be devoted to the above aspect. For bringing RV1 to practice, it is relevant to include other vehicles and traffic flows, such as public transport, bicycles and pedestrians. These additional traffic flows do not hamper the use of RV1 policies. In addition, multiple criteria, like vehicle speed and CO2 emissions, can be included. This chapter has shown that insights and approximations obtained using an MDP framework are of practical importance to improve the control of traffic lights.

Acknowledgements This work has been funded by grants from the Spanish state (TIN2015-66680-C2-2-R) and, Junta de Andalucía (P11-TIC-7176), in part financed by the European Regional Development Fund (ERDF).

Appendix: Notation

This section summarizes the notation used for defining the MDP: i.e. for respectively, the state, action, transition probability, and value function.

$\mathbf{s}$	$= (l, \mathbf{q} = (l, q_1, q_2, \ldots, q_F)$, with $l \in L$ = state of traffic light, and $q_f \in \{0, 1, \ldots, Q-1\}$ is # cars queued at lane f
F	= Number of traffic flows (=lanes)
$a \in A(\mathbf{s}) \subseteq L$	= Change of traffic lights in state $\mathbf{s}$
$P(\mathbf{q}'\|\mathbf{s}, a)$	$= \prod_{f=1}^{F} p_f(q'_f\|(l, q_f), a)$ = Probability that next period's state is
$\mathbf{s}'$	$= (a, \mathbf{q}')$, when current state is $\mathbf{s}$ and action a is taken
$p_f(q'_f\|(l, q_f), a)$	= Single-period transition probability related to lane f.
λ	= probability a car arrives at lane f
$c(\mathbf{q})$	= Costs per period
$v^*(\mathbf{s})$	= Relative valuation of state s of an optimal policy
g^*	= Gain of Markov chain for an optimal policy = Long-run average costs per period of an optimal policy
$\pi^*(\mathbf{s})$	= Optimal change of traffic lights in state $\mathbf{s}$

References

1. L. Baskar, B. De Schutter, J. Hellendoorn, Z. Papp, Traffic control and intelligent vehicle highway systems: a survey. IET Intell. Transp. Syst. **5**(1), 38–52 (2011)
2. S. Bhulai, Dynamic routing policies for multiskill call centers. Probab. Eng. Inf. Sci. **23**(01), 101–119 (2009)
3. R. Haijema, Solving large structured Markov decision problems for perishable inventory management and traffic control. Ph.D. thesis, Univeristy of Amsterdam, Tinbergen Institute, Amsterdam School of Economics, 2008
4. R. Haijema, E.M. Hendrix, Traffic responsive control of intersections with predicted arrival times: a Markovian approach. Comput. Aided Civ. Infrastruct. Eng. **29**(2), 123–139 (2014)
5. R. Haijema, J. van der Wal, An MDP decomposition approach for traffic control at isolated signalized intersections. Probab. Eng. Inf. Sci. **22**(4), 587–602 (2008)
6. J.F. Herrera, E.M. Hendrix, L.G. Casado, R. Haijema, Data parallelism in traffic control tables with arrival information, in *Euro-Par 2014: Parallel Processing Workshops* (Springer, Berlin, 2014), pp. 60–70
7. H.X. Liu, A. Danczyk, Optimal sensor locations for freeway bottleneck identification. Comput. Aided Civ. Infrastruct. Eng. **24**(8), 535–550 (2009)
8. J.Y.K. Luk, Two traffic-responsive area traffic control methods: SCAT and SCOOT. Traffic Eng. Control **25**, 14–22 (1984)
9. M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, Y. Wang, Review of road traffic control strategies, in *Proceedings of the IEEE*, vol. 91 (IEEE, New York, 2003), pp. 2043–2067
10. W.B. Powell, Approximate Dynamic Programming: Solving the Curses of Dimensionality. Wiley Series in Probability and Statistics (Wiley, New York, 2007)
11. M.L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (Wiley, New York, 2014)
12. A. Stathopoulos, L. Dimitriou, T. Tsekeris, Fuzzy modeling approach for combined forecasting of urban traffic flow. Comput. Aided Civ. Infrastruct. Eng. **23**(7), 521–535 (2008)
13. J. Wijngaard, Decomposition for dynamic programming in production and inventory control. Eng. Process. Econ. **4**, 385–388 (1979)
14. D. Zhao, Y. Dai, Z. Zhang, Computational intelligence in urban traffic signal control: a survey. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. **42**(4), 485–494 (2012)

Chapter 14
Smart Charging of Electric Vehicles

Pia L. Kempker, Nico M. van Dijk, Werner Scheinhardt, Hans van den Berg, and Johann Hurink

Abstract A crucial challenge in future smart energy grids is the large-scale coordination of distributed energy generation and demand. In the last years several Demand Side Management approaches have been developed. A major drawback of these approaches is that they mainly focus on realtime control and not on planning, and hence cannot fully exploit the flexibility of e.g. electric vehicles over longer periods of time.
In this chapter we investigate the optimization of charging an electric vehicle (EV). More precisely, the problem of charging an EV overnight is formulated as a Stochastic Dynamic Programming (SDP) problem. We derive an analytic solution for this SDP problem which in turn leads to a simple short-term bidding strategy. From an MDP point of view this solution has a number of special features:

- It leads to analytic optimal results based on order statistics.
- It allows for a more practical rule which can be shown to be nearly optimal.
- It is robust with respect to the modeling assumptions, showing little room for further improvement even when compared to a solution with perfect foresight.

P.L. Kempker (✉)
TNO, Cyber Security & Robustness TNO, The Hague, The Netherlands
e-mail: pia.kempker@tno.nl

N.M. van Dijk
Stochastic Operations Research, University of Twente, Enschede, The Netherlands
e-mail: n.m.vandijk@utwente.nl

W. Scheinhardt • J. Hurink
Department of Applied Mathematics, University of Twente, Enschede, The Netherlands
e-mail: w.r.w.scheinhardt@utwente.nl; j.l.hurink@utwente.nl

H. van den Berg
TNO, Cyber Security & Robustness TNO, The Hague, The Netherlands

Department of Applied Mathematics, University of Twente, Enschede, The Netherlands
e-mail: j.l.vandenberg@tno.nl

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_14

Numerical results with real-world data from the Belgium network show a substantial performance improvement compared to standard demand side management strategies, without significant additional complexity. (This chapter is based on Kempker et al. (Proceedings of the 9th EAI international conference on performance evaluation methodologies and tools, Valuetools 2015, Berlin, 14–16 December 2015, pp 1–8, 2016).)

14.1 Introduction

Due to the limited availability of fossil fuels and their greenhouse effect, a change of our energy supply to a generation based on renewable sources like wind and sun is needed. This shift towards a sustainable energy supply is already ongoing and called the 'energy transition'. However, this change has a drastic impact on the control of the energy system, as generation based on renewable sources is fluctuating and uncontrollable, leading to a decrease of flexibility on the production side of the supply chain. Therefore, the energy transition urges for more flexibility in other parts of the energy supply chain. "Smartening" the grid and transforming the domestic customers from static consumers into "smart users" in the production process can help to overcome these issues.

The use of flexibility on the consumer side (or demand side) of the energy supply chain is called Demand Side Management (DSM; see e.g. [6]). In literature, management methodologies are proposed to exploit the flexibility of consuming devices. One of these methodologies is the so-called PowerMatcher (PM), which reflects a recognized technology and aims to integrate demand and supply flexibility in the operation of the electricity system by dynamic pricing in combination with a hierarchical bidding procedure, see e.g. [4, 9]. However, as the PowerMatcher only focuses on the coordination of demand and supply on the short-term, it has problems to fully exploit the flexibility of shiftable demands over longer periods of time (e.g. for charging electric vehicles) and to achieve the efficiency potentially attainable due to this flexibility.

Recently, the PowerMatcher has therefore been extended by a planning module, which provides coordinated predictions of demands and prices over longer periods of time, e.g. a full day ahead. The output of the planning module can then be used as input for users to determine their short-term bids in order to:

- meet their targets (e.g. fully charge an electric vehicle in time); and
- minimize charging costs.

In this chapter we refer to the extended PowerMatcher as EPM.

An important aspect for the proposed extension is to exploit the output of the planning module for establishing a short-term bidding strategy for a user with a certain amount of shiftable demand over the longer term (e.g. one night). Dealing with the lack of any additional information, we make the modeling assumption that prices for the short-term intervals covered by the planning period are independent and identically distributed according to a normal distribution with mean value equal to the long-term average. We show that finding the optimal short-term bidding

strategy using this assumption can be formulated as an MDP/SDP problem. This MDP/SDP problem can in principle be solved numerically. In fact, with an independence assumption over multiple time periods, even an explicit analytic solution can be derived. This in turn leads to a simple short-term bidding strategy. We also briefly argue how the MDP/SDP approach can still be used for situations with more dependence in the stochastic (fluctuations of the) prices during the planning period. Numerical results show the performance of the Extended PowerMatcher (EPM) compared to the 'standard' PowerMatcher and various other charging strategies. In particular, using simulations with detailed real-world data for wind production and household demands over a 6 months period, it is shown that the PowerMatcher extension with the simple bidding strategy works very well and in many cases provides a considerable improvement over the standard PowerMatcher.

Finally, it appears that the outcomes of the EPM (despite the modeling assumptions mentioned above) are surprisingly close to the (theoretical) optimum achievable even when all realized prices would have been known in advance (to be referred to as strategy 6 in Sect. 14.4). The latter result implies, at least within the present framework, that more precise estimations of the prices for successive short-term time intervals are not imperative since the algorithm is sufficiently robust w.r.t. the modeling assumptions. As such MDP/SDP has shown to be of practical use for future implementations

- at network level for offering bidding strategies,
- at consumer (EV) level to minimize the daily cost for electric driving.

The chapter is organized as follows. In Sect. 14.2, some background on DSM and a short description of the PowerMatcher and its planning module extension is presented. Next, in Sect. 14.3, we concentrate on the charging problem for a single user (EV). First we briefly present the steps for obtaining a practical rule. Next an MDP/SDP formulation and its analytic solution are presented leading to an explicit short-term bidding strategy and a heuristic for the EPM. Section 14.4 provides the numerical validation and evaluation based on weather data for the country of Belgium, by comparison between the DP-heuristic and simple charging strategies. Section 14.5 contains conclusions and topics for further research.

14.2 Background on DSM and PowerMatcher

A vast amount of literature on smart energy systems concerns DSM methodologies [1, 4, 5, 7]. These methodologies generally focus on strategies to optimize the consumption pattern of consumers and to exploit the potential of distributed generation and electricity storage systems. The methodologies optimize at a device level, where the considered smart devices are consuming, producing or buffering devices (e.g. electric vehicles, PV panels or batteries). These smart devices can have multiple options for their operation at every moment in time, of course respecting the device specific constraints (e.g. a fridge can be switched on or delayed, modulating boilers have multiple run levels and an energy buffer can be (dis)charged with

a certain amount of energy). The control methodologies take a decision for every controllable device, often in a hierarchical way to maintain scalability. Furthermore generic functions are used to express the device constraints and differences between the control options of the devices. These functions are called utility functions, bidding functions, cost functions, etc and reduce the optimization problem to a cost optimization problem with a limited number of constraints. The outcome of such

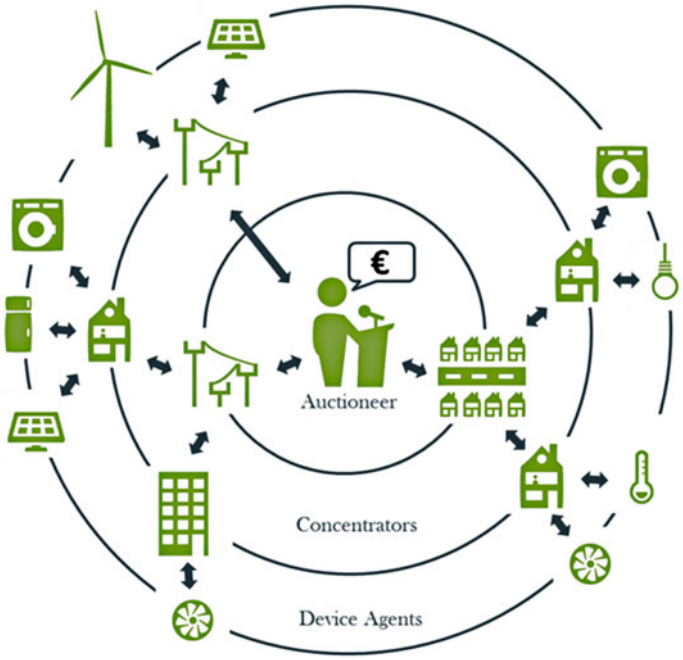


Fig. 14.1: Visualization of PowerMatcher concept (source: http://flexiblepower.github.io/technology/powermatcher/)

an optimization problem is an energy price, which in combination with the cost function of the devices specifies the resulting dispatch for each device.

The demand side management methodologies differ in the way how time is considered. Some only use real-time control, i.e. they only optimize for the decisions at the current moment, whereas other management methodologies optimize for a longer period, often combined with a real-time control to compensate for forecasting errors. For the real-time control strategies mostly cost functions in combination with an optimization methodology are used. These optimization methodologies can be split up in three groups: auction based optimization, game theory based optimization and other mathematical optimization strategies. Planning strategies determine the best sequence of decisions for every point in time on beforehand. However, changes in the environment alter the value of the decision points. In other words,

cost functions in a certain time interval may depend on decisions taken in earlier time intervals. In literature, a lot of auction based methodologies for real-time control are proposed. In an auction based strategy the generators and consumers of electricity place bids for the energy price (=cost function), these are aggregated and then a market clearing price is calculated. One of the more developed methodologies in this area is the PowerMatcher (see [4] and Fig. 14.1 for more details).

A drawback of real-time approaches is that they decide only for the next time moment whereas many devices offer their demand flexibility over longer time horizons. For the optimal utilization of this flexibility, a coordinated use of the devices based on predictions over longer periods of time (e.g. day-ahead or week-ahead horizons) is essential: Electric Vehicles (EVs) for example, can shift their demand within the charging window between being plugged in and leaving again.

In the following, we sketch a coordinated planning extension to the PowerMatcher. The planning module uses information from the household devices along with forecasts of e.g. the wind power generation to predict the average price over the following long-term period, and then make this prediction available to all agents for use in their short-term strategies. This is achieved by combining two instances of the PowerMatcher algorithm, using different time scales:

- *Long-term scale (planning module):* Each agent sends a demand profile for the long-term horizon (total expected demand vs. average price). The planning agent then predicts the average price over this time horizon.
- *Short-term scale (matching module):* Each agent sends its current demand profile for the short-term horizon. Shiftable devices (e.g. electric vehicles) can make their short-term demand flexibility dependent on the long-term average price. The matching agent then determines the market-clearing price for the next short-term interval.

If predictions about the fluctuation of the demands within the time window are available, the same architecture can be used to estimate the standard deviation of the price over a time period T. This is the case for a numerical example in Sect. 14.4. Detailed wind power predictions can be derived from the weather forecast, and the aggregate fixed household demands follow a predictable daily pattern, from which a standard deviation is easily derived. A more detailed description of the extended version of the PowerMatcher can be found in [3].

For implementing the long-term scale, we apply a moving-horizon approach, recomputing the expected average price for the updated time window on a regular basis. In the later sections, where we focus on charging electric vehicles, we use a night-ahead time horizon with a fixed end time, since in the scenarios we consider, the EV has to be charged until the next morning. For other shiftable devices other long-term horizons might be preferable, e.g. a 24-h-ahead moving horizon for battery storage.

In the following sections we consider the planning and control problem resulting from the introduction of the two different phases in more detail and discuss how we may use the extra information provided by the coordinated planning mechanism. We do this for the example of an electric vehicle in a network powered by wind turbines and a diesel generator.

14.3 Optimal Charging Strategies

In order to develop the long-term bidding strategy, the PowerMatcher is confronted with the following question: Given the predictions of the long-term average price and its standard deviation, as provided by the planning module of the PowerMatcher, how can this knowledge be used in the creation of short-term demand profiles?

Since different types of shiftable devices have different objectives and constraints, in this chapter we focus on the particular case of an EV that needs to be charged for up to a certain amount overnight. More precisely, our objective is:

Objective: How to charge an electric vehicle overnight if the prices for the different time periods during the night are uncertain.

Finding an appropriate bidding function for a specific time period means that we need to find an appropriate amount of energy to be charged during that period *for each possible value* of the price. Thus, even though we do not know the value of the price beforehand, we can treat it as given, and then concentrate on the objective above, where future prices are uncertain, but the current price is known.

In this section we approach this objective in a number of conceptual steps based on MDP/SDP in order to determine an optimal charging strategy. These steps are as follows:

Steps:

- Assumptions
- An MDP/SDP formulation
- An analytic solution based on order statistics
- A more practical rule or DP-heuristic rule
- A computational comparison of the DP-heuristic and simple rules.

First, in Sect. 14.3.1 the problem is formulated as a general MDP/SDP problem. Here we use the modeling assumption that prices for different periods are independent in order to obtain a computationally attractive solution form. This assumption is reflected upon later in Sect. 14.4 by numerical results. Next, in Sect. 14.3.2, we focus on the special case of i.i.d. prices; i.e., we now also assume that different periods have the same price distribution, as is in principle the case in the context of the extended PowerMatcher, where we only know the long run average price. For this special case the MDP/SDP is shown to have an analytic solution. Finally, since the explicit solution is based on order statistics, which are in general not easy to compute, in Sect. 14.3.3 we present a simple heuristic which is more practical to implement within the PowerMatcher. Both the modeling assumptions and the heuristic are justified by simulations presented in Sect. 14.4.

For illustrative purposes, we consider a concrete setting of an EV which has to be charged with 8 kWh of energy overnight (8 pm–8 am), and which is connected to a network including wind and fossil fuel power.

14.3.1 MDP/SDP Problem Formulation

If the energy prices were known for each period, then the charging problem could be regarded as a deterministic Knapsack Problem (see any introductory OR book for standard Knapsack formulations). However, from a single user's point of view, these prices are not known in advance: They depend on external factors, such as weather conditions, and on other users' demand profiles.

A stochastic modeling approach is therefore proposed to incorporate the complex bidding and price setting process. Accordingly, we introduce random variables P_t for the price per unit of energy in period t, where $t = 1,\dots,T$. Later on, in the numerical experiments in Sect. 14.4, we choose suitable distributions for these, with their expectation matching the long run expected average price from the planning module. For presentational convenience from now on we consider these prices to be given in discrete units (i.e., they are represented by discrete random variables).

Our objective now is to minimize the total cost to charge an EV within T time intervals (e.g. for a one-night period, $T = 24$ periods of half an hour). At times $t = 1,2,\dots,T$ a decision has to be made how much is to be charged during period t (i.e., within time interval $[t,t+1)$). Let

P_t be the stochastic price variable for period t,
L be the total amount of energy to be charged by the end of period T (i.e. during periods $1,2,\dots,T$),
x_t be the amount still to be charged at time t, (i.e. during periods $t,\dots,T$),
u_t be the amount of energy to be charged during period t (decision variable),
$u_{\max}$ be the maximum amount of energy that can be charged during any period.

As visualized in Fig. 14.2 for a simple example, the charging problem inherently has the structure of a Decision Tree, or more precisely of a Markov Decision Problem (MDP) (see [8] for an extensive treatment of MDPs) with the following repetitive decision structure: Given the state at a particular time t, a decision is to be taken, due to which:

- There are immediate expected costs for period t.
- The resulting state at time $t+1$ is determined stochastically.

Our goal is to determine optimal decisions u_t which depend on the actual state at time t. This state description should contain sufficient information to determine the next state by a stochastic description. For our application, as opposed to standard knapsack formulations, this means that the state description should not only contain the amount x_t to be charged, but also the ('known') current price p_t. Hence, the state at time t needs to be given by $s_t = (x_t; p_t)$.

Let $V_t(x;p)$ be the minimal expected cost during time intervals $t,\dots,T$, given that the state at time t is $(x;p)$ (i.e., given that we need to charge x during $\{t,\dots,T\}$ and that the current price is $P_t = p$).

The optimal decisions for all states can be determined by iteratively solving the following Stochastic Dynamic Programming equations: First, for $t = T+1$ we have

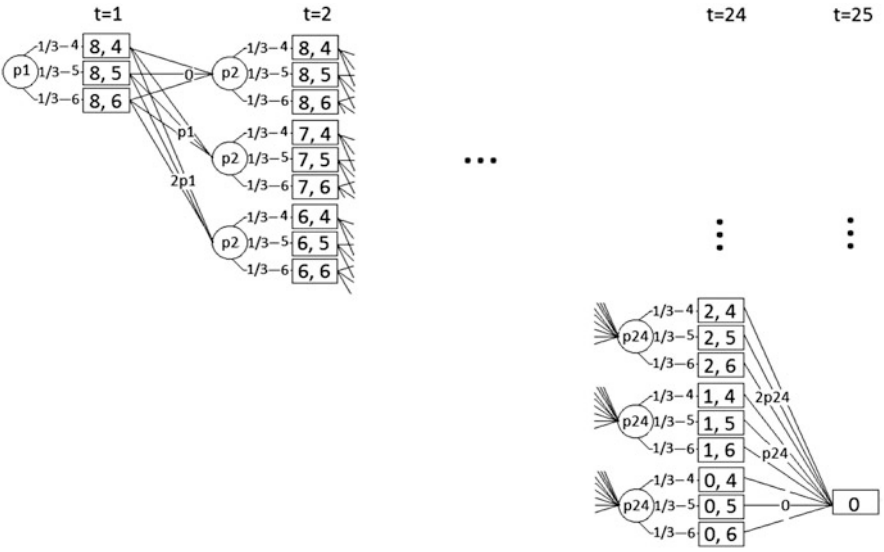


Fig. 14.2: Decision tree example for charging 8 kWh over 24 h, with a discrete price distribution $\mathbb{P}(P_t = p) = 1/3$ for $p = 4,5,6$, and possible charging decisions $u_t \in \{0,1,2\}$

$$V_{T+1}(x) = \begin{cases} 0 & \text{if } x = 0 \\ \infty & \text{otherwise.} \end{cases} \tag{14.1}$$

Next, for $t = T, T-1, \ldots, 1$ and any $(x; p)$ we have

$$V_t(x;p) = \min_u \Big[up + \sum_{p'} \mathbb{P}(P_{t+1} = p') V_{t+1}(x-u; p') \Big]. \tag{14.2}$$

Here the sum is taken over all p' in the support of P_{t+1}, and the minimum over all u in $[0, u_{\max}]$. Finally, the charging problem with an amount of L to be charged over $[1, T+1]$ and with price p at $t = 1$ is denoted by $V(L; p)$. The decision variable (or 'action', or 'control') u which minimizes (14.2) is given by $u_t(x; p)$ and can be said to provide an optimal strategy. Thus, formally we define $u_t(x; p)$ as the amount we need to charge at time t to minimize the total expected cost, when we need to charge x during $\{t, \ldots, T\}$ and $P_t = p$.

Returning to Fig. 14.2, this figure illustrates the example given earlier with $T = 24$, $L = 8$, $u_{\max} = 2$, and $P_1, \ldots, P_{24}$ i.i.d. with price distribution $\mathbb{P}(P_t = p) = 1/3$ for $p = 4,5,6$.

Remark 14.1 (Penalty Costs). Instead of a strict charging requirement at the end of period T, we can also implement a penalty function for a remaining amount not charged, e.g. by a fixed and a proportional penalty for $x_{T+1} = x_T - u_T$ by

$$M \mathbb{1}_{x_{T+1} > 0} + x_{T+1} b.$$

Remark 14.2 (History-Dependent Prices). The formulation of our MDP/SDP can easily be adapted for the case of history-dependent prices. In that case the state of the system should include, apart from x_t and p_t, also the price history $h_t = (p_1, \ldots, p_{t-1})$. Furthermore, the probabilities in (14.2) need to be replaced by history-dependent conditional probabilities, so that (14.2) becomes

$$V_t(x_t; p_t, h_t) = \min_u \big[u p_t + \sum_{p'} \mathbb{P}(P_{t+1} = p' | P_t = p_t, h_t) V_{t+1}(x_t - u; p', h_{t+1}) \big].$$

However, the joint distribution of $P_1, P_2, \ldots, P_T$ generally is not available (since this would require much more communication than provided in the extended PowerMatcher) and even if it is, the computations are computationally prohibitive or require a special approximate procedure.

By the independence assumption and the MDP/SDP approach, optimal strategies and corresponding values can now be computed directly and be numerically implemented in the PowerMatcher. Since we only have information available about the long-run average price, we assume that all prices are i.i.d. with this expectation. In the next section we see that in this case an explicit analytic optimal decision rule can be given, with its corresponding minimal cost. Note that real-world energy prices are in general not i.i.d., but follow daily and seasonal patterns. However, due to the robustness of the PowerMatcher algorithm, possible performance improvements due to more accurate models are limited (see Sect. 14.4).

14.3.2 Analytic Solution for i.i.d. Prices

In this section, we assume that the prices $P_1, \ldots, P_T$ are independent and identically distributed (i.i.d.) random variables. The particular choice of the underlying distribution is not of interest in this context, but for the PowerMatcher its expectation should match the long-run expected average prices resulting from the long-term bidding process.

Recall the definition of $V_t(x; p)$, and $u_t(x; p)$, and the MDP/SDP formulation in (14.2). To find explicit expressions for V_t and u_t for $t = 1, \ldots, T$ we use the concept of order statistics, defined as follows.

Considering the prices $P_t, \ldots, P_T$, we denote the k-th smallest value of these by $P^t_{(k)}$ for $k = 1, \ldots, T - t + 1$. Thus, we have

$$P^t_{(1)} \le P^t_{(2)} \le \cdots \le P^t_{(T-t+1)}.$$

In particular $P^t_{(1)}$ and $P^t_{(T-t+1)}$ are respectively the minimum and maximum prices during $\{t, \ldots, T\}$. For notational convenience we also define $P^t_{(0)} = 0$ and $P^t_{(T-t+2)} = \infty$. When some prices coincide, the ordering is not unique, but this is not important

since we only need their values in the sequel. In fact we only need the expected values of the order statistics, but note that these depend on the whole price distribution.

An optimal strategy and a corresponding value function are given in the following theorem.

Theorem 14.1. *Let* $k = \lfloor \frac{x}{u_{\max}} \rfloor$. *An optimal strategy for* (14.2)*, in the case of i.i.d. prices, is given by*

$$u_t(x;p) = \begin{cases} 0 & \text{if } p > \mathbb{E}P_{(k+1)}^{t+1} \\ x - ku_{\max} & \text{if } \mathbb{E}P_{(k)}^{t+1} < p \le \mathbb{E}P_{(k+1)}^{t+1} \\ \min(u_{\max}, x) & \text{if } p \le \mathbb{E}P_{(k)}^{t+1} \end{cases} \tag{14.3}$$

and its corresponding value function (minimal cost) is

$$\begin{aligned} V_t(x;p) &= \infty \quad \text{if } \frac{x}{u_{\max}} > T - t + 1, \text{or else} \\ V_t(x;p) &= u_{\max} \sum_{\ell=1}^{k} \mathbb{E}[P_{(\ell)}^t | P_t = p] + (x - ku_{\max})\, \mathbb{E}[P_{(k+1)}^t | P_t = p]. \end{aligned} \tag{14.4}$$

Intuitive explanation. Suppose we have perfect knowledge of all future prices within the time horizon under consideration: Then we would pick the $k = \lfloor \frac{x}{u_{\max}} \rfloor$ cheapest intervals out of the time horizon of $T - t + 1$ intervals, and charge $u_{\max}$ during each of these intervals. If x is not a multiple of $u_{\max}$ then we would charge the remainder $x - ku_{\max}$ in the next-cheapest interval available.

However, in the setting of this chapter we do not have perfect knowledge of all future prices: At each time step, we only know the current price (or a set of possibilities for the bidding function) and (an estimate of) the probability distribution of the future prices. This means we have to work with expectations instead: At time t we decide to charge $u_{\max}$ if we expect the current given price to be among the k cheapest prices within the next $T - t + 1$ time intervals.

Proof. W.l.o.g. we let $u_{\max} = 1$ by an appropriate choice of units. We prove optimality of the strategy given in (14.3) by showing that this strategy and the corresponding expression (14.4) for V satisfy the Dynamic Programming equation in (14.2). We proceed by induction, as follows.

Base step ($t = T$). The constraint that a total of x needs to be charged by time $T + 1$ translates to

$$V_{T+1}(x;p) = \begin{cases} \infty & \text{if } x > 0 \\ 0 & \text{if } x = 0 \end{cases}$$

for all p. Using this, the minimizing argument from (14.2) shows that the strategy for $t = T$ satisfies $u_T(x;p) = x$ for $x \le 1$, which is in accordance with (14.3) since $\mathbb{E}P_{(1)}^{T+1} = \infty$ (minimum of an empty set). Also, for $x > 1$ we can choose any u as the minimizing argument in (14.2), so the strategy given in (14.3) is indeed a valid optimal strategy. Furthermore, from (14.2) we also find that

$$V_T(x;p) = \begin{cases} \infty & \text{if } x > 1 \\ up & \text{if } x \le 1, \end{cases}$$

which satisfies statement (14.4) for $t = T$.
Induction step. Supposing that statements (14.3) and (14.4) hold for $t+1$ and $u_{\max} = 1$, we need to prove that (14.3) and (14.4) also hold for t. We define the auxiliary function $v(u)$ as the function to be minimized in (14.2) and rewrite it as

$$\begin{aligned} v(u) &= up + \mathbb{E}_{P_{t+1}}\Big[\sum_{\ell=1}^{\lfloor x-u\rfloor} \mathbb{E}[P_{(\ell)}^{t+1}|P_{t+1}] + (x-u-\lfloor x-u\rfloor)\mathbb{E}[P_{(\lfloor x-u\rfloor+1)}^{t+1}|P_{t+1}]\Big] \\ &= up + \sum_{\ell=1}^{\lfloor x-u\rfloor} \mathbb{E}P_{(\ell)}^{t+1} + (x-u-\lfloor x-u\rfloor)\mathbb{E}P_{(\lfloor x-u\rfloor+1)}^{t+1}, \end{aligned} \tag{14.5}$$

where the expectation $\mathbb{E}_{P_{t+1}}$ is w.r.t. P_{t+1}, the other expectations in the first line are w.r.t. $P_{t+2},\dots,P_T$, and the expectations in the last line are w.r.t. $P_{t+1},P_{t+2},\dots,P_T$. To minimize $v(u)$, note that

$$\lfloor x-u\rfloor = \begin{cases} \lfloor x\rfloor, & \text{for } u \in [0, x-\lfloor x\rfloor] \\ \lfloor x\rfloor - 1, & \text{for } u \in (x-\lfloor x\rfloor, 1] \end{cases}, \tag{14.6}$$

and hence

$$\begin{aligned} \frac{\partial}{\partial u} v(u) &= p - \mathbb{E}P_{(\lfloor x-u\rfloor+1)}^{t+1} \\ &= \begin{cases} p - \mathbb{E}P_{(\lfloor x\rfloor+1)}^{t+1} & \text{if } u \in [0, x-\lfloor x\rfloor] \\ p - \mathbb{E}P_{(\lfloor x\rfloor)}^{t+1} & \text{if } u \in (x-\lfloor x\rfloor, 1] \end{cases}. \end{aligned}$$

Since $v(u)$ is continuous w.r.t. $u \in [0,1]$ and $\mathbb{E}P_{(\lfloor x\rfloor+1)}^{t+1} \ge \mathbb{E}P_{(\lfloor x\rfloor)}^{t+1}$ there are three cases:

- If $p > \mathbb{E}P_{(\lfloor x\rfloor+1)}^{t+1}$ then $\frac{\partial v}{\partial u} > 0$ for all $u \in [0,1]$ and hence the minimum is attained at $u = 0$.
- If $\mathbb{E}P_{(\lfloor x\rfloor)}^{t+1} < p \le \mathbb{E}P_{(\lfloor x\rfloor+1)}^{t+1}$ then $\frac{\partial v}{\partial u} \le 0$ for $u \in [0, x-\lfloor x\rfloor]$ and $\frac{\partial v}{\partial u} > 0$ for $u \in (x-\lfloor x\rfloor, 1]$, and hence the minimum is attained at $u = x-\lfloor x\rfloor$.
- If $p \le \mathbb{E}P_{(\lfloor x\rfloor)}^{t+1}$ then $\frac{\partial v}{\partial u} \le 0$ for all $u \in [0,1]$, and the minimum is attained at $u = 1$ (or $u = x$ if $x < 1$).

This proves the optimality of (14.3). Now we can prove (14.4):

$$
\begin{aligned}
V_t(x;p) &= \min_u v(u) \\
&= \begin{cases} \sum_{\ell=1}^{\lfloor x \rfloor} \mathbb{E}P_{(\ell)}^{t+1} + (x - \lfloor x \rfloor)\mathbb{E}P_{(\lfloor x \rfloor+1)}^{t+1} & \text{if } p > \mathbb{E}P_{(\lfloor x \rfloor+1)}^{t+1} \\ (x - \lfloor x \rfloor)p + \sum_{\ell=1}^{\lfloor x \rfloor} \mathbb{E}P_{(\ell)}^{t+1} & \text{if } \mathbb{E}P_{(\lfloor x \rfloor)}^{t+1} < p \le \mathbb{E}P_{(\lfloor x \rfloor+1)}^{t+1} \\ p + \sum_{\ell=1}^{\lfloor x \rfloor-1} \mathbb{E}P_{(\ell)}^{t+1} + (x - \lfloor x \rfloor)\mathbb{E}P_{(\lfloor x \rfloor)}^{t+1} & \text{if } p \le \mathbb{E}P_{(\lfloor x \rfloor)}^{t+1} \end{cases} \\
&= \sum_{\ell=1}^{\lfloor x \rfloor} \mathbb{E}\left[P_{(\ell)}^{t} | P_t = p\right] + (x - \lfloor x \rfloor)\mathbb{E}\left[P_{(\lfloor x \rfloor+1)}^{t} | P_t = p\right]
\end{aligned}
$$

□

The optimal strategy given here is an almost bang-bang type strategy, in the sense that for x_t being any multiple of $u_{\max}$, the optimal decision $u \in [0, u_{\max}]$ appears to be either 0 or $u_{\max}$.

14.3.3 DP-Heuristic Strategy

The implementation of the optimal strategy in (14.3) requires us to compute expectations of order statistics. This is a difficult task in terms of computational complexity. Therefore we propose an appealing heuristic, which may replace the optimal rule, as follows. With $k = \lfloor \frac{x}{u_{\max}} \rfloor$, let the heuristic strategy be given by

$$
u_t(x;p) = \begin{cases} \min(u_{\max}, x) & \text{if } F(p) \le \frac{k}{T-t+1} \\ x - k u_{\max} & \text{if } \frac{k}{T-t+1} < F(p) \le \frac{k+1}{T-t+1} \\ 0 & \text{if } F(p) > \frac{k+1}{T-t+1}, \end{cases} \tag{14.7}
$$

where F is the common price distribution function. At first sight this may seem equivalent to the optimal rule in (14.3): For strictly monotone F we have

$$
p \le \mathbb{E}P_{(k)}^{t+1} \quad \Leftrightarrow \quad F(p) \le F\left(\mathbb{E}P_{(k)}^{t+1}\right).
$$

$F\left(P_{(k)}^{t+1}\right)$ is a random variable which is distribution is given by the distribution of the (k)-th order statistic of $T-t$ uniformly distributed and independent (i.i.d.) random variables on $[0,1]$, so $\mathbb{E}F\left(P_{(k)}^{t+1}\right) = \frac{k}{T-t+1}$. However the heuristic strategy (14.7) is not equivalent to the optimal strategy (14.3), since

$$
F\left(\mathbb{E}P_{(k)}^{t+1}\right) \neq \mathbb{E}F\left(P_{(k)}^{t+1}\right)
$$

in general.

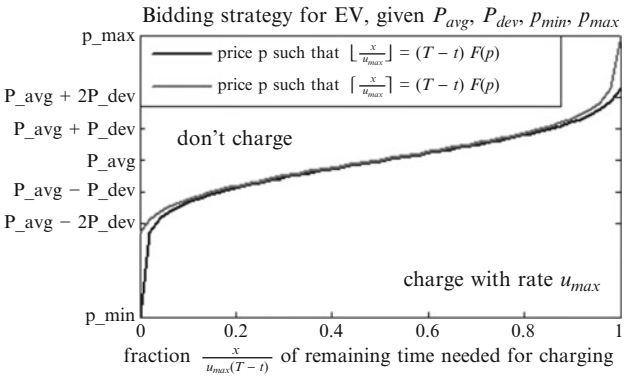


Fig. 14.3: Bidding/charging strategy of EV, with the threshold DP-heuristic strategy given by (14.7)

In what follows the heuristic based on (14.7) is referred to as DP heuristic. Even though the difference between the optimal and heuristic DP rules may not be negligible, we still use it in the next section with good results, thus strengthening our belief that the EPM is a strong concept which seems to be robust with respect to the strategy that is used.

The heuristic bidding strategy is illustrated in Fig. 14.3. The distance between the thresholds corresponding to $\lfloor \frac{x}{u_{\max}} \rfloor$ and $\lceil \frac{x}{u_{\max}} \rceil$ (left and right entries) depends on x_0 and $u_{\max}$ (in Fig. 14.3, $x_0 = 8$ and $u_{\max} = \frac{1}{2}$).

14.4 Numerical Results

In this section, we compare the performance of different short-term charging strategies in simulations using real-world data for wind production and fixed household demands.

The data we use is available at [2]. It consists of 15-min averages for the network of Belgium over the first 180 days of 2015 for:

- the measured load, aggregated over all households and industries (we consider this the fixed demand),
- the wind production, measured in some locations and upscaled to fit the network's nominal capacity,
- long-term predictions of the aggregated load and of the wind production, updated once a day at 11 am for all 15-min intervals in the next 24 h.

The Belgian network includes 4.8% wind production and 3.3% solar production: We exclude solar production here since the only shiftable devices considered in the simulation are electric vehicles. These charge overnight when the solar production is zero. To compensate, and in order to incorporate the sharp increase in wind pro-

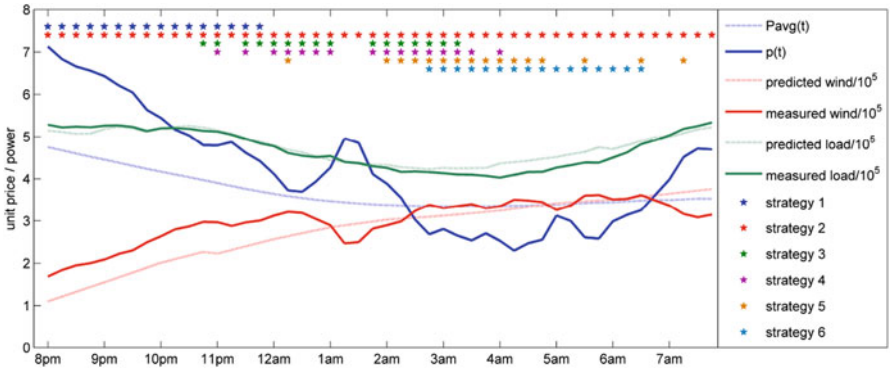


Fig. 14.4: Comparison of different bidding/charging strategies, prices and demand/production for night 124

duction over the recent and coming years, we scale the overall loads by a factor of $\frac{1}{5}$, while keeping the wind power as it is.

For the purposes of this simulation, electric vehicles can charge from 8 pm till 8 am, and their demand is fixed at 8 kWh/night. For an analysis of realistic driving patterns, we refer to [10].

In our example network, power is produced by wind turbines (generating cheap electricity whenever there is wind) and a diesel generator (which has virtually unlimited capacity but high unit prices). We add one EV to the network, in addition to the fixed loads provided in the data. Since the demand of the EV is very small compared to the fixed load, the EV is a *price taker*: while the current energy price influences the EV's decisions, the EV's actions have no significant influence on the prices. This would change if we added many EVs with similar demand patterns (e.g. one for every household), a possible topic for further research.

We compare the following strategies for charging the EV:

Strategies:

1. Charge directly when plugged in, as fast as possible,
2. Charge evenly over the whole night,
3. DP-heuristic strategy (14.7), using last night's average and standard deviation as P_{avg} and P_{dev} (this corresponds to the standard PowerMatcher architecture),
4. DP-heuristic strategy (14.7), using coordinated night-ahead estimates of P_{avg} and P_{dev}, estimated once at the beginning of the night (this corresponds to the PowerMatcher approach with only one prediction round per night),
5. DP-heuristic strategy (14.7), using coordinated night-ahead estimates of P_{avg} and P_{dev}, updated every 15 min (this corresponds to the extended PowerMatcher approach)
6. Lower bound strategy, where we pick the cheapest time slots in retrospect, assuming perfect knowledge or individual estimates of all prices.

Strategy	Single night		180 nights	
	Cost	Improvement	Cost	Improvement
1	45.3	–	10,861	–
2	33.4	26%	10,672	1.7%
3	30.5	33%	10,626	2.2%
4	28.5	37%	10,190	6.2%
5	23.9	47%	9971	8.2%
6	22.0	51%	9909	8.8%

Table 14.1: Cost of charging during night 124 (*left*) and total cost of charging for nights 1–180 (*right*). Percentual improvements are w.r.t. the simple (fast charging) strategy 1; recall that strategies 3, 4 and 5 are based on the DP-heuristic expression (7), while strategy 6 assumes full knowledge

With unit prices $p_{\text{wind}} = 1$ and $p_{\text{diesel}} = 10$, we find the average unit price at time t by calculating

$$p_t = \frac{p_{\text{wind}} * \text{wind used} + p_{\text{diesel}} * \text{diesel used}}{\text{wind used} + \text{diesel used}}.$$

In Fig. 14.4, the outcomes of the different strategies are illustrated for night 124 of 2015. All strategies lead to different charging times, as indicated by $\star$ in the figure. The costs for charging 8 kWh in this particular night can be found in the left part of Table 14.1. Compared to charging when plugged in (strategy 1), the standard PowerMatcher (strategy 3) leads to 33% lower costs. The coordinated planning algorithm introduced in Sect. 14.2 (strategy 5) leads to another significant improvement (47% lower cost than strategy 1), and performs only 4% worse than the lower bound (strategy 6).

The total costs for charging one EV every night of the first 180 nights of 2015 can also be found in Table 14.1, on the right panel. Some nights show no difference in costs (e.g. because there is no wind, and hence the unit prices are constant), while in other nights the difference is large (as in night 124). In total, over the first half of 2015 and using imperfect price predictions, the standard PowerMatcher (strategy 3) performs 2.2% better than strategy 1. Another improvement of 4% can be realized by running the PowerMatcher once at 8 pm (strategy 4). Using the PowerMatcher every 15 min instead of once at 8 pm (strategy 5) reduces the costs by another 2%, with only 0.6% difference compared to the lower bound of 9909.

Remark 14.3 (Independence Assumption). Note that in Sect. 14.3 we assumed that the distribution $F(p)$ is known: In the simulations we modeled the prices as i.d.d. and normally distributed for the EV strategy. Even though the realizations of the prices resulting from the Belgian data are neither i.i.d. nor normally distributed, the algorithm still leads to large cost reductions, with little room for further improvement. This illustrates the robustness of the PowerMatcher algorithm w.r.t. modeling simplifications and seems to justify the modeling assumption of i.i.d. prices.

14.5 Conclusion/Future Research

In this chapter we have shown the value of MDP/SDP for a future extension of the PowerMatcher for coordination of distributed demand and supply in smart energy grids. This classical SDP approach and its detailed analytic results enable a smart exploitation of the flexibility of demand and prices for e.g. charging EVs. Also a more practical rule could be developed which turned out be nearly optimal. The effectiveness of the simple-to-use strategy for charging EVs was verified numerically by using real-world data.

One future step is to include 'dependent' demands over consecutive periods, such as by using time serial (econometric) methods. Hereby it is of interest to investigate if similar structural results still remain obtainable and if extended numerical procedures can be developed.

Another interesting next step is to include so many EVs that their combined decisions have a significant influence on the price. Other straightforward topics for further research are the extension of the methods used in Sect. 14.3 to other shiftable devices (e.g. battery storage), or other types of controllable devices. A remaining topic of interest is the inherent trade-off between the amount and type of information provided by a planning module and its additional computation and communication.

Acknowledgements We acknowledge the contributions of Yvonne Prins, Pamela MacDougall, Koen Kok, and Leon Kester (all affiliated with TNO) to the research leading to this chapter.

Appendix

Energy Terminology

Demand Side Management (DSM): Use of the flexibility of electric devices in households or offices to influence the electricity demand profile.

Shiftable devices: Electricity consuming devices, for which the demand of electricity can be shifted in time. Examples are washing machines, dryers and dishwashers.

Electric vehicles (EV): Vehicles like e.g., cars, busses, trucks or bicycles, which use electricity from batteries to drive or to support driving.

PowerMatcher: A DSM technology using dynamic pricing and a hierarchical bidding process to support the matching of supply and demand of electricity.

Notation

S	State space (discrete/continuous)
T	Number of periods, time $t = 0, 1, ..., T-1$
$s_t = (x_t, p_t)$	State, with:
x_t	Amount still to be charged after time t
p_t	Price per unit of energy at time t
$a_t = u_t$	Action/amount to be charged at time t
$c^a(s) = u_t p_t$	Expected one step cost in state $s_t = (x_t, p_t)$, under charge action $a = u_t$
$A(s_t) : u \leq u_{\max}$	Set of actions available in state s_t
δ_t	Decision rule at time $t : u_t(x_t, p_t)$
$P(j\|(s;a))$	$\mathbb{P}(x_t - u_t\|(x_t, p_t))$ One step transition probability under action $u_t(x_t, p_t) = u_t$ in state $s_t = (x_t, p_t)$
$V_t(s)$	$V_t(x_t, p_t)$ Optimal value function of expected cumulative costs over remaining $T-t$ steps up to time T, starting in state $s_t = (x_t, p_t)$ at time t

References

1. C. Block, D. Neumann, C. Weinhardt, A market mechanism for energy allocation in micro-CHP grids, in *41st Hawaii International Conference on System Sciences* (2008), pp. 172–180
2. ELIA, Data download (2015). http://www.elia.be/nl/grid-data/data-download
3. P. Kempker, M. van Dijk, W. Scheinhardt, J. van den Berg, J. Hurink, Optimization of charging strategies for electric vehicles in powermatcher-driven smart energy grids, in *Proceedings of the 9th EAI International Conference on Performance Evaluation Methodologies and Tools, Valuetools 2015*, Berlin, 14–16 December 2015 (2016), pp. 1–8
4. K. Kok, The PowerMatcher: smart coordination for the smart electricity grid. Ph.D. thesis, Vrije Universiteit, Amsterdam, 2013
5. A. Molderink, V. Bakker, M. Bosman, J.L. Hurink, G.J.M. Smit, Management and control of domestic smart grid technology. IEEE Trans. Smart Grid **1**(2), 109–119 (2010)
6. A. Molderink, V. Bakker, J.L. Hurink, G.J.M. Smit, Comparing demand side management approaches, in *IEEE PES Innovative Smart Grid Technologies Conference (ISGT Europe)* (2012), pp. 1–8
7. J. Oyarzabal, J. Jimeno, J. Ruela, A. Englar, C. Hardt, Agent based micro grid management systems. in *International Conference on Future Power Systems 2005. IEEE* (2005), pp. 6–11
8. M.L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (Wiley, New York, 2014)

9. S. Rafiei, A. Bakhshai, A review on energy efficiency optimization in smart grid, in *38th Annual Conference on IEEE Industrial Electronics Society (IECON)* (2012), pp. 5916–5919
10. V. Silva et al., Estimation of innovative operational processes and grid management for the integration of EV. Project deliverable D6.2 (2011)

Part IV
Production

Chapter 15
Analysis of a Stochastic Lot Scheduling Problem with Strict Due-Dates

Nicky D. van Foreest and Jacob Wijngaard

Abstract This chapter considers admission control and scheduling rules for a single machine production environment. Orders arrive at a single machine and can be grouped into several product families. Each order has a family dependent due-date, production duration, and reward. When an order cannot be served before its due-date it has to be rejected. Moreover, when the machine changes the production of one type of family to another family, a setup time is incurred. The problem is to find long-run average optimal policies that accept or reject orders and schedule the accepted orders.
To obtain insight into the optimal performance of the system we model it as a Markov decision process (MDP). This formal description leads to, at least, three tangible goals. First, for small scale problems the optimal admission and scheduling policy can be obtained with, e.g., policy iteration. Second, simple heuristic policies can be formulated in terms of the concepts developed for the MDP, i.e., the states, actions and (action-dependent) transition matrices. Finally, the simulator required to study the performance of heuristic policies for large scale problems can be directly implemented as an MDP. Thus, the formal description of the system in terms of an MDP has considerable off-spin beyond the mere numerical aspects of solving the MDP for small-scale systems.

15.1 Introduction

In this chapter we use Markov decision theory to model a production-to-order environment in which a single bottleneck machine produces one product family at a time and is subject to significant setup times when the product family changes.

N.D. van Foreest (✉) • J. Wijngaard
Faculty of Economics and Business, University of Groningen, Groningen, The Netherlands
e-mail: n.d.van.foreest@rug.nl; j.wijngaard@rug.nl

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_15

The sequence in which families can be produced is arbitrary, rather than, for instance, cyclic. Orders for all families arrive with geometrically distributed inter-arrival times, require deterministic service time and have strict due-dates. If an arriving order cannot be produced before its due-date, it has to be rejected upon arrival. The aim of the production system is to maximize the long run average reward per unit time. Following the survey in [1] we refer to this problem as a *customized stochastic lot scheduling problem (CSLSP)* with strict due-dates. Here, the term *customized* refers to make-to-order characteristic of the production situation, contrary to e.g. the Stochastic Economic Lot Sizing Problem (SLSP) in which production is make-to-stock hence allows more freedom in deciding the moment, type and quantity of items to produce.

The CSLSP finds its motivation in a wide array of batch-processing environments, such as the production of paper, release liners, or paint, in which customer orders have to be grouped in families with similar chemical of physical properties, and in which the switch-over times from one family to another are relatively long. In these settings, suppliers not only have to ensure short and reliable delivery times to their customers, but also have to realize high machine utilization due to the extremely large operational costs of the equipment. More specifically, a thorough industrial motivation is provided in [2]. The problem to efficiently produce baseball bats is discussed in [3]. The authors of [4] concentrate on the scheduling of packaging pharmaceuticals on order. Finally, the first author has applied the ideas developed in related work, c.f. [5, 6], to improve scheduling rules at paper mills and release liners.

Clearly, maximizing the long run expected reward per unit time requires a policy that judiciously accepts orders and schedules the accepted orders. Designing good policies is, however, quite difficult since the acceptance and scheduling decisions are subtlety related: the acceptance depends on a schedule of previously accepted orders, and if the order is accepted, the order should be scheduled such that all orders in the schedule are produced before their due-date. To illustrate, consider the policy that separates each two accepted orders by a setup and applies a FIFO service discipline. This policy is certainly simple, but also unattractive as it leads to a low acceptance rate. Intelligent policies should try to form 'runs' of contiguous orders of the same family to reduce the fraction of time spent on setups. In fact, it may sometimes be better to reject an order even when the schedule allows to accept it, the idea being that given the content of the current schedule the rejection leaves room for later, more attractive, arrivals so that longer runs can be formed.

The contribution of this chapter is to show how MDPs can be used to model and analyze the CSLSP. Since the CSLSP is a (very) difficult scheduling problem and lacks much simplifying structure, e.g., linearity, the theory of MDPs seems to be one of the few, if not the only, methods that can be used to analyze this problem. The formalization of the CSLSP in terms of an MDP is, admittedly quite an elaborate task, but it has considerably off-spin beyond just the computation of an optimal policy. The formalized concepts, i.e., actions, rewards and transition matrices, enable us to specify heuristic policies in a mathematical unambiguous way. As a consequence, we can benchmark heuristic policies against the optimal policy within one framework, i.e., in terms of MDPs, for not too-large scaled problems. Also, with

the MDP specification we can set up simulations for large scale systems. Finally, the formal structure facilitates the communication about the system's (desired) behavior, so that the debugging of the simulation code becomes easier, and, once the project is finished, the specification serves as documentation. Thus, a second aim of this chapter is to showcase the multifarious usefulness of MDPs.

The work presented here is based on previous work of the authors. The authors of [7] develop a number of simple heuristic scheduling and acceptance policies and use simulation to analyze and compare the performance of these heuristics. In [8] and [2] simulation is also used to analyze aspects of the CSLSP. Reference [6] contains the first implementation of the CSLSP as an MDP to compute optimal policies and benchmark the heuristics developed earlier in [7]. Finally, the authors of [5] extend the MDPs to more complicated systems with, e.g., family-dependent due-dates, and use ideas developed in [9] to improve the heuristic policies, again with the tools of MDPs.

The chapter is organized as follows. In Sect. 15.2 we discuss work related to the CSLSP. Section 15.3 presents the model of the production system, the admissible policies, and the optimization problem. Section 15.4 describes the associated Markov decision process and formulates an interesting heuristic policy in terms of the concepts developed for the MDP. Finally, Sect. 15.5 shows some results to illustrate the model. The last section concludes.

15.2 Theoretical Background of the CSLSP

Although variations of the CSLSP with due-dates have been investigated previously, the analysis by means of MDPs seems not to be addressed before in the literature.

Polling models, see for instance [10] or [1], consider problems that can be seen as relaxations of our production situation, for at least three reasons. First, many polling systems are concerned with unlimited queues, while the restriction to meet due-dates in our situation puts a natural limit on the length of the schedule. Polling models that do consider finite queue sizes, for instance [11, 12], relate a single queue to each product family. In our case, however, the queueing capacity can be fully shared among all families. Second, numerous polling models that include setups, e.g., [13–15], assume that the server incurs a setup even when it visits an empty queue. This is a reasonable assumption for polling models of telecommunication systems since the locations between the queues are physically separated and the server (token) has to visit a queue to see whether it is empty or not. However, setting up a machine for non-present families appears quite unnatural for the scheduling problems occurring in manufacturing situations; planners usually have a complete view of the order portfolio. Third, many polling models, see e.g., [16], are concerned with cyclic policies. However, our earlier work, c.f. [7], shows that it is detrimental to cyclically serve the product families for the CSLSP with due-dates; the heuristic threshold policy performs at least a few percent better. Strategies that use fixed polling tables, such as developed in [17], also seem hard to apply here. As the due-dates of the orders

in the production system need to be respected, the content of the schedule is highly dynamic. Moreover, the schedules we are concerned with may contain *multiple* runs of one product family. This last aspect has a further consequence in that policies such as, e.g., 'serve a queue of orders of the same family to exhaustion', see for instance [18], are not suitable here. It may be necessary to switch service from one family to another before all orders of the family in service are produced.

Also the approach developed in [19] is not suitable for our case. They consider fixed group sizes for orders with the same family and optimize over the sizes of the groups. However, in the resulting policy group, completion times are not guaranteed, hence orders may not meet their due-dates.

15.3 Production System, Admissible Policies, and Objective Function

We start with describing the production system. Next, in Sect. 15.3.2, we introduce the decisions that govern the acceptance and scheduling of arriving orders. Section 15.3.3 discusses the objective function.

15.3.1 Production System

The production situation is modeled as follows. A single server receives a stream of orders at arrival epochs $0 = T_0 \leq T_1 \leq T_2, \ldots$. The inter-arrival times $T_{i+1} - T_i$ are i.i.d., integer valued, and geometrically distributed with success parameter p, that is, according to

$$P(T_{i+1} - T_i = k) = p(1-p)^k, \quad \text{for } k = 0, 1, 2, \ldots.$$

Since $E(T_{i+1} - T_i) = (1-p)/p$, the arrival rate $\lambda = p/(1-p)$. Arriving orders belong to family f, which is one of N possible families, with probability q_f, independent of anything else. Thus, the arrival rate of family f is $\lambda_f = \lambda q_f$. A job of family f—we use the word 'job' and 'order' interchangeably—requires b_f time units of service, where b_f is deterministic and integer valued. An order can be rejected upon arrival, but if accepted it is scheduled for service such that it can be produced before its due-date. We define the *due-date horizon* of an accepted job as the time left to its due-date, and we say that an order is *tight* whenever its horizon is equal to the sum of the order sizes and setup times scheduled in front of it together with its own service time. Upon arrival each jobs has an initial due-date horizon of length h, which is the same for all families. We remark in passing that, while it is possible in the model to make the initial due-date horizons family-dependent, there is not a direct practical relevance. In industry, see, e.g., [2], it is standard practice to use a uniform lead time, as it is not so clear how to exploit different due-date horizons. Most surely

the family with the shortest horizon will often be denied access to machine capacity, and in general, this form of unfairness appears to be quite undesirable and hard to resolve when different horizons are allowed.

Whenever two subsequent orders in the schedule belong to different product families a setup of (integer valued) duration u, which is the same for all families, is inserted between these two orders. When an order arrives at an empty system, a setup is not necessary if the last produced order is of the same family as the arriving order.

We also require that service of orders and setups is non-preemptive.

With regard to the scaling of the system, observe that without loss of generality it is possible to scale the due-date horizon and the sizes of the jobs and setups such that at least for one family the jobs have unit time length or the setup time is of unit length.

15.3.2 Admissible Actions and Policies

We now describe the actions that determine the acceptance/rejection of orders and the formation of job schedules. In principle many actions are available: jobs can be rejected or accepted, after a service completion the machine can stop, even though there is work in the queue, accepted jobs can be inserted anywhere in the schedule as long as the due-date constraints are satisfied, and so on. Some thought reveals, however, that only three actions are interesting to consider. These admissible decisions, *Combine*, *Spawn*, and *Reject*, are deterministic (only dependent on the state of the schedule). We illustrate these actions below by means of the schedule instance in Table 15.1. With these actions we form admissible policies, which we take to be stationary and non-anticipative with respect to the arrival process and work-conserving (non-idling if the schedule contains orders).

1	2	3	4	5	6	7	8	9	10	11	12	13	14
R	R	s	B	B	B	s	G	s	R				

Table 15.1: An instance of a schedule of accepted orders. The symbols ‘R’, ‘G’, and ‘B’, refer to order colors, ‘s’ to a setup. The size of the schedule is 14 positions. Jobs and setups have unit length here

The *Combine* action aims at reducing the fraction of setups by trying to combine a new order with a *run* of the same family, i.e., a set of consecutive orders of the same ‘kin’. By simple pairwise interchange arguments, see e.g., [20], it is easy to see that the optimal sequence of jobs within a run is Earliest Due-Date first—the reward cannot increase by inserting arriving orders before orders of the same family—hence, the Combine action adjoins accepted orders only to the *end* of a run. To illustrate, suppose a ‘blue order arrives. The schedule in Table 15.1 contains one blue run, that is, at positions 4–6. The Combine action tries to join the order

with this run by inserting it at position 7 and shifting the already present orders at positions 8 and 10 back to positions 9 to 11. In case either one of the orders at positions 8 and 10 will be late as a result of the insertion, Combine is not allowed to insert this new order at position 7.

The *Spawn* decision adjoins a new order to the end of the schedule so that it 'spawns' a new generation of its family. This acceptance will necessarily introduce a setup between the last family and the new order. Hence, Spawn schedules the just arrived job at position 12 and inserts a setup at position 11. Thus, to accept such an order the schedule should at least provide room for the setup and the order. If the content of the schedule is such that both Combine and Spawn are admissible, the former action is always chosen. From a practical point of view this is entirely reasonable: why delay an order more than necessary?

The *Reject* decision simply rejects the arriving order.

It is evident that many more actions are possible, but it seems that the above actions are the most interesting. For instance, it might be possible to spawn a new run in between two other runs, but there does not seem to be a good reason to do so: the due-date horizon of the new run is larger than the run that comes after it. Likewise, when trying to combine a new arrival, the only sensible choice is to combine it with the last run of the same family in the schedule. Otherwise the horizons of larger number of runs will increase. To see this, assume that in the schedule of Table 15.1 a red order arrives. This might be combined with the red orders in positions 1 and 2, or with the order at position 10. Even if it would be possible to combine the new order with the first run, it simply makes no sense to consider it.

15.3.3 Objective Function

The last step of the model specification consists of formulating a reward structure. We set the acceptance reward for a job of family f equal $r_f > 0$ and the earliness cost to zero. The underlying motivation is that we assume that serving orders early is acceptable when this potentially leads to accepting more orders. Hence, the reward for accepting an order must be higher than the cost of producing early. We also set the setup cost equal to zero. However, inserting setups arbitrarily cannot be optimal, since a setup takes away a position in the schedule thereby preventing potential rewards. There is no tardiness cost, as jobs cannot be late.

The objective is to find the maximal long run average reward per unit time of the production system. More formally, given policy π, let $A^\pi(t)$ be the sum of the rewards of the accepted orders up to and including time t. Then the long run average expected reward per unit time, $J(\pi)$, takes the form

$$J(\pi) = \lim_{t\to\infty} \frac{E\left(A^\pi(t)\right)}{t}. \tag{15.1}$$

The objective is to find

$$J^\star = \max_\pi J(\pi), \tag{15.2}$$

where the maximization is taken over the class of admissible policies. Observe that in case the reward per job is equal to the job length, the reward function $J(\pi)$ represents the long run average fraction of accepted work, i.e., the utilization, associated with policy π.

As an aside, the objective is often formulated in terms of a limit inferior. However, since the state space is finite, this subtlety is unnecessary as the limit exists by Proposition 8.1.1 of [21].

15.4 The Markov Decision Process

We now turn to modeling the above production situation as a Markov decision process (MDP). An MDP consists of a set of states, a set of decisions, state transition functions $P(s'|s,a)$ representing the transition probability from state s to a future state s' conditional on choosing decision a, and a function $c(s,a)$ that provides the reward earned when taking decision a in state s. For general background on the definition and analysis of Markov decision processes we refer to [21] or [22].

As the specification of the system state s is somewhat involved, we characterize the format of a state in Sect. 15.4.1. The actions have already been introduced in Sect. 15.3.2; we only need to formalize the actions, which we do in Sect. 15.4.2. Section 15.4.3 presents the state transition functions. The rewards $c(s,a)$ are already clear from Sect. 15.3.3: only decisions that lead to the acceptance of an order of family f generate positive reward r_f. In Sect. 15.4.4 we discuss a suitable method to aggregate problem instances for which job size, arrival rate and reward are identical for all families as this allows us to extend the system sizes considerably. Then, in Sect. 15.4.5 we explain a convenient method to generate the state space of the Markov chain. Finally, in Sect. 15.4.6 we use the formulation developed earlier to specify the heuristic policy that performed best in [7].

15.4.1 Format of a State

We now turn to a characterization of the states,

Observe that, as the decision epochs coincide with the arrival epochs, we only have to specify the state at the arrival epochs.

The simplest, but naive, state description would be to represent the schedule by means of a tuple of jobs $(j_1, j_2, \ldots j_n)$, and the ith job in the schedule as a tuple (f_i, h_i) where f_i is the job's family and h_i its horizon. Thus, the schedule of accepted jobs would written as $(f_1,h_1),\ldots,(f_n,h_n)$. Besides the schedule of jobs, the state should also include the family f of the arriving job, so that the state becomes

$$((f_1,h_1),(f_2,h_s),\ldots,(f_n,h_n);f)\,.$$

Note that from this sequence it is trivial to find the setups, hence it is not necessary to include these.

This detailed state description is, however, unsuitable for numerical purposes: the state space very quickly becomes unmanageable large. For this reason we develop a way to aggregate the states by reconsidering the minimal information the actions require, as only this information need to be captured in the state description.

First, observe that for the Spawn decision it is only necessary to know the length of the schedule and the family of the new job. Second, the Combine decision requires the due-date horizon of the 'tightest' order of each run, i.e., the order with the smallest due-date horizon, since if the tightest order of a run is in time, all orders in the run will be in time. Finally, the Reject decision can be applied to any state, regardless the content of the schedule or the family of the arriving order. Thus, it suffices that a state contains the sequence of runs, and, per run, the family, the length (including a setup), and the due-date horizon of the 'tightest' job. As it is straightforward to convert the due-date horizon to a due-date horizon of the first job in the run, we will henceforth assume that the first job in the run is also the tightest job.

To cast the above in suitable notation, write $s = (\sigma; f)$ where σ denotes the content of the schedule and f indicates the family of the arriving order. The schedule σ is an ordered tuple of runs $(\sigma_1, \ldots, \sigma_n)$. A run σ_i is also an ordered tuple (f_i, h_i, r_i), where f_i is the run's family, h_i the due-date horizon of the first job (hence the tightest job) in the run, and r_i the length (including setup time).

The first run σ_1 needs no due-date horizon information, since maintaining a run's horizon is only relevant to ensure that earlier runs cannot become too long. As there are no runs in front of σ_1, the horizon can be dropped, thus, $\sigma_1 = (f_1, r_1)$. When the schedule is empty, we write $\sigma = (f_1, 0)$, where f_1 is the last produced family.

A simple observation allows us to reduce the information in s still further. Since arriving orders cannot be combined with runs in front of a tight run, nor can new runs be spawned before this tight run, the family and horizons of all the runs in front of the last tight run in the schedule are superfluous, hence the lengths of these runs can be added simply to the length of this tight run, and this run can be taken as the run in the schedule. More formally, suppose that the kth run is the last tight run. Then the schedule $\sigma = (\sigma_1, \ldots, \sigma_n)$ can be further simplified to the schedule $\sigma' = (\sigma'_1, \sigma'_2, \ldots,)$, where $\sigma'_1 = (f_k, \sum_{i=1}^{k} r_i)$, $\sigma'_2 = \sigma_{k+1}$, and so on.

Finally, in case the new job is combined with a run in the schedule, we need to project back the horizon of the newly accepted job to the horizon of the first, i.e., tightest, job of the run. Suppose that the arriving order with due-date horizon h is to be combined with run σ_k. As the due-date horizon of the first job is h_k, its processing time should start at $h_k - b_{f_k}$ time units from now at the latest, and the setup should start at $h_k - b_{f_k} - u$ time units from now. Hence, as the run will then be finished at $h_k - b_{f_k} - u + r_k$, the new job cannot start earlier than this time. Clearly, to ensure that also the new job will be in time, the due-date horizon of the first job must be such that

$$h_k - b_{f_k} - u + r_k \leq h - b_{f_k},$$

as $h - b_{f_k}$ is the latest possible start of the new job. Therefore, h_k being the horizon of the first job before the acceptance, the horizon of the first job must be set to

$$h'_k = \min\{h_k, h - (r_k - u)\}, \tag{15.3}$$

after the acceptance.

15.4.2 Actions and Operators

Making a transition from one state to the next involves three steps. The first step is to apply one of the actions Reject, Spawn, and Combine to the arriving order. The second step consists of generating the time to the next arrival epoch, and removing the related amount of workload from the schedule. In the third step, a new arriving order is generated. We implement each step by means of operators, to be defined now. For notational convenience, let $|\sigma| = \sum_{i=1}^{n} r_i$ denote the total amount of work in the schedule including setups.

First we associate the actions Reject, Spawn and Combine to operators $\mathscr{R}, \mathscr{S}, \mathscr{C}$ which map *states* $s = (\sigma; f)$ to *schedules* $\sigma = (\sigma_1, \ldots, \sigma_n)$. The operator $\mathscr{R}$ simply removes the arriving order:

$$\mathscr{R}(\sigma; f) = \sigma;$$

S adjoins the order to the end of the schedule,

$$\mathscr{S}(\sigma; f) = \left(\sigma_1, \ldots, \sigma_n, (f, h, u + b_f)\right),$$

where u denotes the length of a setup and b_f the size of the new job, and the operator $\mathscr{C}$ combines the arrival with the last run of family f in the schedule:

$$\mathscr{C}(\sigma; f) = (\sigma_1, \ldots, \sigma_{k-1}, (f, h'_k, r_k + b_f), \sigma_{k+1}, \ldots, \sigma_n), \tag{15.4}$$

where h'_k is defined in (15.3), and we assume that σ_k is the last run of family f in the schedule.

For each state s, the set $A(s)$ contains the actions that can be applied to s. Specifically, since the Reject action is always possible, $\mathscr{R} \in A(s)$ for all s. With regard to the Combine action, $\mathscr{C} \in A(s)$ if the insertion of the new of family f by Combine does not lead to any other job in the schedule becoming late. In more formal terms, suppose σ_k is the last job of family f in σ, as in (15.4). Then the insertion is allowed if $h_i \geq b_f + \sum_{l=1}^{i} r_l$ for all $i \geq k+1$. Finally, $\mathscr{S} \in A(s)$ when $\mathscr{C} \notin A(s)$ and $|\mathscr{S}(s)| \leq h$, i.e., the length of the schedule that results after the operation of $\mathscr{S}$ on s is less than or equal to h.

We next need the *time shift* operator $\mathscr{T}$, which maps a *schedule* to a *schedule*, to implement the effect on the schedule when the time to the next arrival epoch is one time unit. Clearly, if the schedule is not empty and the time to the next arrival epoch

is one time unit, (part of) the first run of the schedule is produced, and the horizon of the remaining runs is reduced by one time unit. Specifically, while temporarily extending the definition of $\mathscr{T}$ to operate on single runs so that we can also write $\mathscr{T}\sigma = (\mathscr{T}\sigma_1, \ldots, \mathscr{T}\sigma_n)$, we have

$$\begin{aligned}
\mathscr{T}(f_1,0) &= (f_1,0),\\
\mathscr{T}\sigma &= (f_1,0), \quad \text{if } |\sigma| = 1,\\
\mathscr{T}\sigma &= (\mathscr{T}\sigma_2, \ldots \mathscr{T}\sigma_n), \quad \text{if } r_1 = 1,\\
\mathscr{T}\sigma_1 &= (f_1, r_1 - 1), \quad \text{if } r_1 \geq 2,\\
\mathscr{T}\sigma_i &= (f_i, h_i - 1, r_i), \quad \text{if } i \geq 2.
\end{aligned} \tag{15.5}$$

Shifting by $l \geq 0$ time units is simple, just apply the l times composition $\mathscr{T}^l$. No shift in time is represented by $\mathscr{T}^0$, which is the identity operator.

The last operator F_f maps a *schedule* to a *state* by augmenting σ with an arriving order of family f:

$$\mathrm{F}_f(\sigma) = (\sigma; f), \quad f = 1, \ldots, N.$$

15.4.3 Transition Matrices

With the above operators we can map a state s to a future state, and hence describe the transition matrices associated with each of the actions Reject, Spawn, and Combine.

If the next order arrives l time units from now and is of family f, then for any $a \in A(s)$ we write $s'_{fl} = (\mathrm{F}_f \circ \mathscr{T}^l \circ a)(s)$ for the next state. The set $F(s)$ of future states of s is given by

$$F(s) = \left\{ s'_{fl} \,\middle|\, s'_{fl} = (F_f \circ \mathscr{T}^l \circ a)(s), \text{ for } a \in A(s), 0 \leq l \leq |a(s)|, 1 \leq f \leq N \right\}, \tag{15.6}$$

recall that $|a(s)|$ denotes the length of a schedule after the operation of a on s). The transition matrices now follow easily: the probability to jump from state s to a state $s'_{fl} \in F(s)$ is

$$P\left(s'_{fl} \,\middle|\, s, a\right) = q_f\, p(1-p)^l, \quad \text{if } l < |a(s)|, \tag{15.7}$$

and

$$P\left(((f_1,0);f) \,\middle|\, s, a\right) = q_f \sum_{l \geq |a(s)|} p(1-p)^l = q_f\,(1-p)^{|a(s)|}, \quad \text{if } l \geq |a(s)|, \tag{15.8}$$

where again f_1 is the last produced family.

15.4.4 Further Aggregation in the Symmetric Case

In the case the families are symmetric, i.e., the arrival rates, job sizes and rewards are equal for all families, it turns out that it is not necessary to store information concerning the family of a run in the schedule. Because of the symmetry in arrival rate, an arriving order is of family f with probability $1/N$. Recall that a new run can only be spawned if it is not possible to combine the arriving order with a run of the same family in the schedule. Since we keep only track of runs in the schedule to which new orders can be combined (i.e. runs preceding a tight run are aggregated, see Sect. 15.4.1) it follows that the number of runs in the schedule cannot exceed the number of families and that each family has at most one run in the schedule. Therefore, the probability that an arriving order sees upon arrival a run in the schedule to which it can be combined is n/N, where n denotes the number of runs in the schedule.

The state can now be aggregated into a tuple $(\sigma;m)$ where $\sigma = (r_1;h_2,r_2;\ldots)$ denotes the content of the schedule and m is an indicator that denotes the run of the schedule to which the arriving order can be combined (in case $m \geq 1$) or the case that no run in the schedule is from the same family as the arriving order (in case $m = 0$). For example, $s = ((2;5,3;8;2);1)$ depicts the state in which an arriving order can be combined with the first run in the schedule.

It is apparent that the operators $\mathscr{C}, \mathscr{R}$, and $\mathscr{S}$ of Sect. 15.4.2 and the state transition probabilities of Sect. 15.4.3 have to be converted to the aggregated process. As this is relatively easy we refrain from including the details.

15.4.5 State Space

The formal characterization of the state space S requires to enumerate all schedules that can be obtained with the operations defined above. this, however, is a cumbersome task due of the interaction between orders and setups. By far the easiest way to generate S is by induction. Using the set of future states of s as defined in (15.6), let $S^{(i+1)} = \bigcup_{s \in S^{(i)}} F(s)$, and take as initial set $S^{(0)} = \{((f,0);g) | f,g = 1,\ldots,N\}$. Once there is an i such that $S^{(i+1)} = S^{(i)}$, it must be that $S = S^{(i)}$. Observe that this iterative procedure stops trivially, due to the finiteness of schedule length and number of different families.

15.4.6 A Heuristic Threshold Policy

With the machinery developed above it is easy to formalize the heuristic policy that was seen to perform best in the simulation studies carried out in [7].

The intuition behind this heuristic is as follows. Any time spent on setups does not increase the reward function as defined in (15.2), and potentially decreases it as setup time cannot be used to process orders. Hence, policies that encourages the formation of long runs, without violating the due-date constraints, appear the most interesting. Furthermore, in case the load of one family does not suffice to fill the machine capacity, it is obvious that the capacity should be shared among a few families, hence, runs of orders of families should alternate. However, as soon as multiple families share the capacity, they are in effect 'competing' for the capacity. As is generally the case, and shown in [7], a good policy should regulate this competition. In fact, we show there that if the load is high and the spawn action is not regulated, typical runs contain just one or two jobs. A good policy should regulate the spawn action to create *combination potential*: runs may only start when the schedule is *not* full so that the first job of a run is *not* tight when the run starts. Since a threshold to enable or disable the spawn action is about the simplest policy possible, we use this threshold in the heuristic policy.

The details of this heuristic are as follows. For a given state s, the threshold policy always applies the combine action if $\mathscr{C} \in A(s)$. If $\mathscr{C} \notin A(s)$, the spawn action is chosen, provided $|\mathscr{S}(s)| \leq ch$, where c is some constant smaller than 1, typically around 0.8. If this condition is not satisfied, the Spawn action is not chosen, and the order will be rejected.

Note that the threshold value c is uniform for the families. We expect this to work well in symmetric cases, that is, in production situations in which the contending families have roughly similar arrival rates, rewards, and job sizes. However, when there is asymmetry, it may become better to make the threshold values family dependent.

15.5 Numerical Study

In this section we carry out a simple performance analysis of the optimal policy as obtained by policy iteration and compare this to the performance of the heuristic policy.[1] We investigate the effect of various system parameters in Sect. 15.5.1. From the results, we will learn that the threshold policy has (near) optimal performance, which makes it interesting to investigate the extent to which the threshold policy resembles the optimal policy. For this purpose we use a method of [9] to visually compare the optimal and heuristic policy.

In this chapter, we restrict the numerical analysis to cases with $N = 3$ families, and initial due-date horizons $h = \{10, 14, 18\}$; we refer to [5, 6] for a much more detailed analysis of the optimal and heuristic policy.

For notational convenience we use vectors b, q and r, to denote the job size, arrival fraction and reward per family. For instance, $b = (b_1, b_2) = (1, 3)$ specifies the scenario in which the job size of the first (second) family is 1 (3). Throughout

[1] We implemented all algorithms in python and numpy, which are freely available on the web. At request the code used for the computation can be obtained from the first author.

we restrict the total load of the system $\rho = \lambda \sum_f q_f b_f$ to the values 0.5, 0.9 and 1.2. Higher or lower values appear less relevant to study from a practical point of view.

15.5.1 Influence of the Load and the Due-Date Horizon

In our first experiment we study the effect of the system load ρ and the due-date horizon h for $N = 3$ families with $b = r = (1,1,1)$ and $q = (1/3, 1/3, 1/3)$. Since $b = r$ the optimal value (15.2) is equal to the utilization of the system. As the performance measure of interest we consider the acceptance ration, i.e., J^*/ρ for the optimal policy and J^H/ρ for the heuristic policy, which we denote by H.

Figure 15.1 contains three panels where $h = 10$ (left), 14 (middle), and 18 (right) to show the influence of h. The straight lines correspond to the acceptance ratio of the optimal policy for ρ is 0.5, 0.9 and 1.4. The variable on the x-axis is the threshold c of the heuristic, so that we can also plot the dependence of J^H/ρ as a function of c.

The figure allows us to make a number interesting observations. First of all, the acceptance ration order decreases as a function of ρ. This is natural: when for instance $\rho = 1.2$ at least 20% of the offered load has to be rejected. Second, since the graphs of the expected rewards of the threshold and optimal policy touch for some value of c, the threshold policy can have (near) optimal performance. Third, when c is large, in the order of 1, the threshold policy performs quite badly. This is due to the fact that it gives rise to schedules in which short runs of jobs alternate with setups, see [7] for further detail. On the other hand, when $c \approx 0$, the performance is also far from optimal. To see this, observe that when runs only start when the schedule is nearly empty, the server must idle quite often. Fourth, the best value of c becomes smaller as ρ becomes larger. This is as expected: the higher the load, the more combination potential by using the combine action, hence longer runs. Finally, by comparing the panels from left to right, to understand the behavior of h, we see that the average reward J increases from left to right. This is not surprising since longer due-date horizons also enable more combination potential.

15.5.2 Visualization of the Structure of the Optimal Policy

From the results of the previous sections it appears that a suitable threshold policy might be optimal which in turn would imply that the optimal spawn/reject decision has a threshold structure. To investigate this observation more formally we follow an approach, proposed in [23], to visualize the structure of the optimal policy by grouping multiple states into the sets

$$\begin{aligned} \mathcal{S}_l^s(a) &= \{s \,|\, \mathscr{S} \in A(s), |\mathscr{S}(s)| = l, \pi^\star(s) = a\}, \quad \text{for } a \in \{\mathscr{S}, \mathscr{R}\}, \\ \mathcal{S}^c(a) &= \{s \,|\, \mathscr{C} \in A(s), \pi^\star(s) = a\}, \quad \text{if } a \in \{\mathscr{C}, \mathscr{R}\}, \end{aligned} \tag{15.9}$$

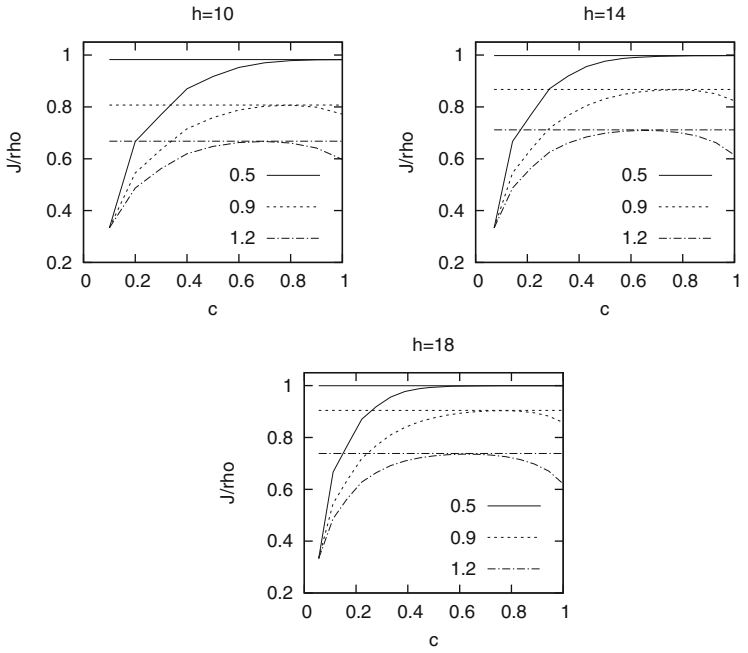


Fig. 15.1: The long run acceptance ratio J/ρ per arrival as a function of the threshold parameter c for the threshold and optimal policy. The *straight lines* correspond to the performance of the optimal policy which, of course, cannot depend on c. The *highest*, *middle* and *lowest* graphs correspond to $\rho = 0.5, 0.9$, and 1.2, respectively. The *left*, *middle*, and *right* panel correspond to $h = 10, 14$, and 18. The other parameters are as follows: $N = 3$, $u = 1$, $b = r = (1,1,1)$, and $q = (1/3, 1/3, 1/3)$

where $\pi^\star(s)$ is the optimal action in state s. For example, $\mathcal{S}_6^s(\mathscr{S})$ is the set of states with $|\mathscr{S}(s)| = 6$ and for which it is optimal to spawn a new run rather than reject the arriving order. Let $\eta^\star(\tilde{\mathcal{S}}) = \sum_{s\in\tilde{\mathcal{S}}} \eta^\star(s)$, where $\eta^\star(s)$ is the steady state fraction of time the $\pi^\star$-controlled MDP spends in state s.

Table 15.2 shows the frequencies $\eta^\star\left(\mathcal{S}_l^s(\mathscr{S})\right)$ (in percentages) and $\eta^\star\left(\mathcal{S}_l^s(\mathscr{R})\right)$ for $N = 3$ and $l = 2, \ldots, 10$. It is clear from the overlap at $l = 7$ that the optimal spawn/reject decision is not of threshold type: for some states with $l = 7$ it is best to spawn while for other states with $l = 7$ it is best to reject. Although this is somewhat disappointing (the simple threshold policy is not optimal), not all structure is lost. If we take the threshold equal to $c = 7$, the heuristic is not optimal in only 2.45% of the states, while it is optimal in the rest, i.e., $100 - 2.45 = 97.55\%$, of the states. Pertaining to the combine/reject decision, the right part of the table shows that the optimal policy always chooses Combine whenever $\mathscr{C} \in A(s)$.

$\lvert\mathscr{S}(s)\rvert$	2	3	4	5	6	7	8	9	10	Total
Spawn	1.04	1.25	2.33	2.86	4.21	3.60	0.00	0.00	0.00	15.29
Reject	0.00	0.00	0.00	0.00	0.00	2.45	8.30	8.00	6.10	24.85

	Total
Combine	51.47
Reject	0.00

Table 15.2: Frequency table for spawn/reject and combine/reject decision. The parameters of the test instance are as follows: $N = 3$, $h = 10$, $\rho = 1.2$, $u = 1$, $b = r = (1,1,1)$, and $q = (1/3, 1/3, 1/3)$

15.6 Conclusion

In this chapter we show how we used Markov decision theory to analyze the customized stochastic lot scheduling problem. The specification of this production system as an Markov decision problem (MDP) proved to be very beneficial to understanding the system for several reasons. First, with the MDP we developed optimal policies for (relatively) small problem instances. With the optimal policy we could evaluate the performance of our earlier developed heuristics against the optimal policy. Next to this, the analysis of the structure of optimal policies provided guidance to improve the heuristics to other, more general situations. Second, the formal specification of the actions and the related transition matrices helped to structure the (coding of the) simulator that we used to study large problem instances. Third, once we had the MDP and the simulator, we could use the results of the MDP to test the output of the simulator for quite large problems, in the order of 10^6 states, thereby providing confidence in the results of the simulator.

We believe that this structured approach extends well beyond the case we studied in this chapter. For instance, small queueing networks with blocking and in which the servers can fail are very hard to implement in a simulation environment. Typically, the design of a simulator for such systems requires the use of finite state machines to structure the possible transitions of the system, prevent deadlocks, and so on. For such situations, the development an MDP can prove to have much offspin. The formulation of the MDP will not only help to understand, decompose, and model the system, but also guide the development of a simulation environment, and finally help to validate the results of a simulator.

Appendix: Notation

$A(s)$	Action state for state s
$A^{\pi}(t)$	Reward of accepted orders until time t under policy π
b_f	Size of order of family f
$\mathcal{C}$	Combine operator

$((f_1,h_1,\ldots,(f_n,h_n))$	Schedule of accepted jobs
c	Fraction of the duedate horizon
h	Due-date horizon of a job
$J(\pi)$	Long-run average reward rate under policy π
J^*	Optimal reward rate
λ	Arrival rate of orders
$\lambda_f = \lambda q_f$	Arrival rate of orders of family f
N	Number of order families
p	Probability of an arrival in a slot
P	Transition function
$\mathcal{R}$	Reject operator
q_f	Probability that an arrival belongs to family f
r_f	Reward for accepting an order family f
$s = (\sigma, f)$	States consisting of schedule σ and arriving order of family f
S	State space
$\|S\|$	Number of elements of the state space
$\mathcal{S}$	Spawn operator
σ	Schedule
$\mathcal{T}$	Time shift operator
T_i	Arrival epoch of order i

References

1. E. Winands, I. Adan, G. van Houtum, Eur. J. Oper. Res. **210**(1), 1 (2011)
2. H. Ten Kate, Order acceptance and production control. Ph.D. thesis, University of Groningen, Groningen, 1995
3. E. Schmidt, M. Dada, J. Ward, D. Adams, Interfaces **31**(3), 16 (2001)
4. L. Strijbosch, R. Heuts, M. Luijten, Int. J. Oper. Prod. Manage. **22**(5), 549 (2002)
5. R. Germs, N. van Foreest, Int. J. Prod. Res. **51**, 940 (2013)
6. R. Germs, N. van Foreest, Eur. J. Oper. Res. **213**, 375 (2011)
7. N.D. van Foreest, J. Wijngaard, J.T. Van der Vaart, Int. J. Prod. Res. **48**, 3561 (2010)
8. J. Bertrand, J. Wortmann, J. Wijngaard, *Production Control, a Structural and Design Oriented Approach* (Elsevier, Amsterdam, 1990)
9. R. Haijema, N. van Dijk, J. van der Wal, C.S. Sibinga, Int. J. Prod. Econ. **121**(2), 464 (2009)
10. H. Takagi, in *Stochastic Analysis of Computer and Communication Systems*, ed. by H. Takagi (Elsevier, Amsterdam, 1990), pp. 267–318
11. H. Takagi, Adv. Appl. Probab. **23**(2), 373 (1991)
12. E. Kim, M. Van Oyen, IIE Trans. **32**(9), 807 (2000)
13. L. Kleinrock, H. Levy, J. Oper. Res. **36**(5), 716 (1988)

14. R. Righter, J.G. Shantikumar, Oper. Res. **12**(3), 146 (1998)
15. D. Markovitz, L. Wein, Oper. Res. **49**(2), 246 (2001)
16. M. Reiman, L. Wein, Oper. Res. **46**(4), 532 (1998)
17. O. Boxma, H. Levy, J. Westrate, Queueing Syst. **9**, 133 (1994)
18. Z. Liu, P. Nain, D. Towsley, Queueing Syst. **11**, 59 (1992)
19. N. Vandaele, I.V. Nieuwenhuyse, S. Cupers, Eur. J. Oper. Res. **151**, 181 (2003)
20. M. Pinedo, *Planning and Scheduling in Manufacturing and Services*, 2nd edn. (Springer, New York, 2008)
21. M. Puterman, *Markov Decision Processes, Discrete Stochastic Dynamic Programming* (Wiley, New York, 1994)
22. H. Tijms, *A First Course in Stochastic Models* (Wiley, Chichester, 2003)
23. R. Haijema, J. Van der Wal, N.M. van Dijk, Comput. Oper. Res. **34**, 760 (2007)

Chapter 16
Optimal Fishery Policies

Eligius M.T. Hendrix, Rene Haijema, and Diana van Dijk

Abstract This paper describes and analyses a bi-level Markov Decision Problem (MDP). The model has been used to study questions on the setting of fisheries quota. The problem extends earlier models in literature and describes fish stock and economic dynamics. At the first level, an authority decides on the quota to be fished keeping in mind long term revenues. At the second level, fishermen react on the quota set as well as on the current states of fish stock and fleet capacity by deciding on their investment and fishery effort. An analysis of the behaviour of the model is given and used to decide on how to discretize the state space. The aim is to derive optimum quota settings based on value iteration. This chapter illustrates how a MDP with continuous state and action space can be solved by truncation and discretization of the state space and applying interpolation in the value iteration.

Key words: Dynamic programming, Fishery, Stochastic programming, Continuous state space, Discretization

E.M.T. Hendrix (✉)
Computer Architecture, Universidad de Málaga, Málaga, Spain
e-mail: eligius@uma.es

R. Haijema
Operations Research and Logistics, Wageningen University, Wageningen, The Netherlands
e-mail: rene.haijema@wur.nl

D. van Dijk
Department of Environmental Social Sciences, Swiss Federal Institute of Aquatic Science and Technology (EAWAG), Dübendorf, Switzerland
e-mail: Diana.vanDijk@eawag.ch

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_16

16.1 Introduction

Authorities in fisheries management are responsible for making decisions in an uncertain context related to the dynamics in fish stock due to environmental variability. From this perspective, it is useful to analyse fishery policies in a setting of stochastic dynamic programming. Moreover, policy makers have to take into account the fisherman behaviour when setting the policy. The policy maker determines quota not only based on known or unknown dynamics in fish stock and fleet capacity of the fishery sector, but also based on how fishermen behave in an unregulated setting. The implementation of these interactions in a model is a challenge, because policy makers and fishermen may have different, and even conflicting objectives. The policy maker seeks a long-term sustainable fish stock as well as maximization of social benefits, while fishermen reveal short term profit seeking behaviour.

Fisheries policies are often analysed in the literature in a setting of dynamic optimization, where a sole owner determines optimal harvest and investment levels for a specific fishery [2, 3, 6, 7]. Such models have been developed in deterministic and stochastic settings, often assuming uncertainty in fish stock dynamics due to environmental variability. The assumption of sole ownership, however, ignores behaviour of fishermen and the extent to which a policy restricts this behaviour. The behaviour of both policy maker and fishermen can be modeled in a game theoretic setting. See [1] for an overview of the application of game theory to fishery over the last 30 years.

In [9, 10], the authors developed a bi-level stochastic dynamic programming model to analyse the interaction between the authority on higher level and the fisheries sector behaviour on lower level. In game theory, such a setting is called a leader-follower (Stackelberg) situation where the optimal policy of the leader is determined by the reaction of the follower, i.e. fishermen behaviour. In the developed model, the objective of a policy maker is to determine levels of quota that maximize social benefits, subject to dynamics in fish stock and capital stock and based on behaviour of fishermen. The model focuses on single-species fisheries, where fish stock growth is considered stochastic. At the lower level the myopic fishermen make their harvest and investment decisions based on available fish stock and capital stock and the restricted quota set by the authority at level one. This approach resembles a two-stage dynamic game.

The resulting model can be considered from the perspective of a Markov Decision Problem (MDP). The challenge to determine the optimal strategy for the authority is now given by the continuous state space of fish and capital stock and the stochastic behaviour of the fish growth. In principle, the solution process can be approached from the viewpoint of Value Iteration (VI). The focus of his chapter is on the truncation and discretization of the state space in this approach.

To investigate this question, we first analyse the behaviour of the model in a deterministic setting. The question is whether the process is stable and converges automatically, whether there are transition states and what are the bounds of the area where the fish and capital stock are developing. Second, we approach the problem from a stochastic perspective within the bounds that have been derived. We argue

that it would be good to base the discretization on the stationary distribution of the state space. As we are dealing with a Markov Decision Problem, the interesting feature is that this distribution depends of course on the optimal policy. However, to derive a policy, we require a choice for the discretization. In this chapter, we illustrate the choice of the discretization and optimal solution with the practical case that has been elaborated in [10].

This paper is organized as follows. Section 16.2 describes the model. Section 16.3 analyses the mathematical convergence behaviour of the model and the natural bounds of the stationary state space. Section 16.4 describes the solution approach based on value iteration and its interaction with the chosen discretization. Section 16.5 summarizes our findings.

16.2 Model Description

The dynamic model is called a bio-economic model, as it describes dynamics of a biological system as well as the dynamics of economic behaviour. The two parts are linked by the decision aspect of the MDP which optimizes an objective function given constraints. We describe the biological dynamics, economic system and optimization separately. In the used symbols, we distinguish between model parameters (exogenous in lower case letters) and decision variables (capitals) that are characterized by direct decision variables, dependent variables and stock variables. Without loss of generality, the dynamics are represented by time steps and using an index t.

16.2.1 Biological Dynamics; Growth of Biomass

The development of the population size of one species of fish S_t measured in kton is based on the Gordon-Schaeffer model. The used parameters are

Data

- m carrying capacity of the specie in kton
- g intrinsic growth rate
- ξ lognormal random variable with c.d.f. $G(\xi)$ based on parameters μ and σ, with $\mu + \frac{1}{2}\sigma^2 = 0$, such that the mean of ξ is 1.

The random variable describes a random multiplicative effect. This makes the lognormal distribution appropriate. Now, let the stock variable be the fish stock S_t in kton and H_t be the harvest in kton, then the dynamics of the fish stock is given by

$$S_{t+1} = S_t + \xi(gS_t\left(1 - \frac{S_t}{m}\right)) - H_t. \tag{16.1}$$

16.2.2 Economic Dynamics; Harvest and Investment Decisions

The economic part of the model describes capital stock dynamics depending on investment in fishery equipment and all costs the sector is confronted with to harvest the fish to be landed and sold. A fixed selling price c^p is assumed. The used parameters are

Data

- c^p selling price in euro/kton
- d yearly depreciation rate of capital
- c^e cost of effort in euro/hpd
- c^i investment cost in euro/hpd
- c^s cost of the sales in euro/euro
- q so-called catchability coefficient in Spence harvest function

Let K_t describe the capital stock and I_t be the investment. Following the neoclassical investment theory we have

$$K_{t+1} = K_t(1-d) + I_t, \tag{16.2}$$

where the fishery sector is confronted with investment costs $c^i I_t$.

To describe the cost of harvesting, a new decision variable E_t is introduced representing the fishing effort (intensity) such that the harvest H_t becomes in fact a dependent variable. The variable E_t is expressed in so-called horse-power-days (hpd), which is a common metric in fishery capital. The relation between harvest H and effort E is one of the elements where the biological and economic model come together. The harvest not only depends on the effort, but also on the current level of fish stock in the sea. Following the Spence harvest function [8]

$$H_t = S_t\left(1 - e^{-qE_t}\right) \rightarrow E_t = \frac{1}{q}\ln\frac{S_t}{S_t - H_t}. \tag{16.3}$$

In the economic submodel, we have that effort is limited by capital

$$E_t \leq K_t \rightarrow H_t \leq S_t\left(1 - e^{-qK_t}\right), \tag{16.4}$$

which implies that harvest is always less than fish stock:

$$H_t < S_t. \tag{16.5}$$

The direct return of the fishery sector of harvest is determined by sales minus its direct cost, $(1-c^s)c^pH_t$. The effort cost c^eE can be expressed in terms of harvest H by substituting variable E: $c^eE = \frac{c^e}{q}\ln\frac{S_t}{S_t-H_t}$. The direct profit for the fishery sector

$$r(\pi_t, S_t, K_t) = (1-c^s)c^pH_t - \frac{c^e}{q}\ln\frac{S_t}{S_t - H_t}$$

depends on the optimal effort and resulting optimal harvest $H_t = H(\pi_t, S_t, K_t)$ decision taken by the fishermen as described in Sect. 16.2.3.

16.2.3 Optimization Model

The objectives to be optimized, depend on the players around the fishery scene, such as different groups of fishing companies, countries and the European Union. In this paper, we focus on an authority that optimizes the discounted stream of future social benefits. Being confronted with a level of fish stock S and of a fleet K, the authority sets quota $\pi(S,K)$ at the first decision level. The fishery sector reacts on that by deciding on investment level $I(\pi,S,K)$ and harvest $H(\pi,S,K)$ given the stock levels and the quota π that was set by the authority at the first level.

16.2.3.1 Decisions at Level 2

At the second level, the harvest decision is limited by quota π, capital stock via (16.4) and the marginal cost. Keep in mind that fish stock can be so low that the effort cost is higher than the return; harvest levels for that year are zero. The profit for the fishery sector given quota π and stock levels S and K is

$$r(\pi,S,K) = \max_{H} \left\{ (1-c^s)c^p H - \frac{c^e}{q} \ln \frac{S}{S-H} \right\}, \tag{16.6}$$

subject to $0 \leq H \leq \pi$ and $H \leq S\left(1-e^{-qK}\right)$. If optimization problem (16.6) has an interior solution, the analytical solution follows from the first order condition

$$\frac{d}{dH} \left\{ (1-c^s)c^p H - \frac{c^e}{q} \ln \frac{S}{S-H} \right\} = 0. \tag{16.7}$$

Given the upper and lower bounds in (16.6), the solution is given by

$$H(\pi,S,K) = \min \left\{ (S - \frac{c^e}{c^p q(1-c^s)})^+, \pi, S\left(1-e^{-qK}\right) \right\}, \tag{16.8}$$

where x^+ stands for $\max\{0,x\}$. With respect to the investment decisions, the model assumes that the fishery sector observes the desired harvest level

$$\hat{h}(\pi,S) := \min \left\{ (S - \frac{c^e}{c^p q(1-c^s)})^+, \pi \right\}) \tag{16.9}$$

and tunes its equipment for next year to have sufficient capital to reach $\hat{h}$

$$\hat{h}(\pi,S) = S_t\left(1-e^{-qK_{t+1}}\right) \rightarrow K_{t+1} = \frac{1}{q} \ln \frac{S_t}{S_t - \hat{h}(\pi,S)}. \tag{16.10}$$

Given the development of capital stock K_t in (16.2) and assuming nonnegativity of investment, this leads to the investment function

$$I(\pi,S,K)=\left(\frac{1}{q}\ln\frac{S}{S-\hat{h}(\pi,S)}-K(1-d)\right)^{+}. \tag{16.11}$$

16.2.3.2 Decisions at Level 1

At the first level, we have that the authority tries to maximize long term welfare

$$\max_{\pi(S,K)} E\sum_{t=0}^{\infty}\frac{r(\pi(S_t,K_t),S_t,K_t)-c^i I(\pi(S_t,K_t),S_t,K_t)}{(1+\alpha)^t}, \tag{16.12}$$

where α is the discount rate and the decision π is subject to the dynamics of fish stock S in (16.1) and capital K in (16.2).

Notice that we are dealing with a stationary system. That means that the optimum strategy consists of a decision rule that tells the authority what quota π to set given fish stock S and capital in fishery equipment K. Furthermore, the optimum strategy depends on the behaviour of the fishery sector at the second level.

16.3 Model Analysis

We address the question what are reasonable values for the decision and state variables given parameter values. Therefore, we first look at implicit bounds in the decision variables and transient states of the model and then consider the model when it is behaving in a stationary way in steady state.

16.3.1 Bounds on Decision and State Space

With respect to the first level, no cost has been introduced so far to set the quota π. This means, one has alternative optimal solutions if π is not binding in the decision on the harvest level. We will always consider the minimum level to be chosen in case π has alternative solutions. Often, this means that $\pi=H$. For convenience, let

$$\hat{s}=\frac{c^e}{c^p q(1-c^s)}$$

denote the level of fish stock under which it is not profitable to start harvesting. Simply, $\pi=H=0$ if $S<\hat{s}$.

The dynamics in fish stock described by (16.1) basically increases up to a carrying capacity m. If the fish stock due to environmental aspects exceeds that level, it can only decrease to m. Due to the fishing behaviour, it is also known that for

$S < \hat{s}$, no fishing takes place and growth is always positive. This means that values $S < \hat{s}$ correspond to transient states. basically, $\hat{s}$ is a lower bound of the fish stock S_t, if the initial stock $S_0 > \hat{s}$. At the other side, if the initial stock is higher than the carrying capacity, $S_0 > m$, then the stock can only go down from that level. So, also values $S > m$ correspond to transient states. For the elaboration of the optimal policy $\pi(S,K)$, given an initial stock S_0, we have that in the system

$$S_t \in [\min\{S_0, \hat{s}\}, \max\{S_0, m\}]. \tag{16.13}$$

This means that the interesting range for harvest H and quota π is $H, \pi \in [0, \max\{S_0, m\} - \hat{s}]$.

These ranges also provide a range of appropriate values for the capital stock K and investment I. Due to investment cost and depreciation, the level of capital should not exceed what is required to catch the desired level, as specified by (16.10); $K, I \in [0, \max\{\frac{1}{q}\ln(\frac{m}{\hat{s}}), K_0\}]$.

16.3.2 Equilibrium State Values in a Deterministic Setting

In this section, we assess the order of magnitude of the decision and state values given an instance of parameter values. In order to get a feeling, the model is considered in a deterministic setting, so $\xi = 1$, where we analyse whether there is a unique absorption state.

In an absorption state, we have that H, I, S, K are constant in time and do not change any more. The harvest is a constant fraction of the fish stock S and in the fish dynamics (16.1), now $S_{t+1} = S_t$. For an absorption state value $S \geq \hat{s}$ one can find

$$H = gS\left(1 - \frac{S}{m}\right) \rightarrow S = \frac{gm + \sqrt{(gm)^2 - 4gmH}}{2g}, \tag{16.14}$$

otherwise no harvest takes place; $H = 0$ and $S = m$.

In the long run, capital and quota levels adapt to the harvest level; $\pi = H$ and $K = \frac{1}{q}\ln\frac{S}{S-H}$ and $I = dK = \frac{d}{q}\ln\frac{S}{S-H}$. The decision maker at level 1 should keep the stationary revenue R as high as possible, whereas harvest equals the growth:

$$R := \max_H \left\{ (1-c^s)pH - \frac{c^e + dc^i}{q} \ln\frac{S}{S-H} \right\}, \tag{16.15}$$

subject to

$$H = gS(1 - \frac{S}{m}).$$

Substitution of the growth in the (constant over time) revenue function gives

$$R = g(1-c^s)c^p S(1 - \frac{S}{m}) + \frac{c^e + dc^i}{q} \ln((1-g) + \frac{g}{m}S). \tag{16.16}$$

For an interior optimum ($S > \hat{s}$), the first order condition of $\frac{dR}{dS} = 0$ yields equilibrium value

$$S = \frac{3m}{4} - \frac{m}{2g}\left(1 - \frac{1}{2}\sqrt{(g-2)^2 + \frac{8g(c^e + dc^i)}{q(1-c^s)mc^p}}\right). \tag{16.17}$$

In the following section, we study how this absorption state value is reached, if the system starts at an arbitrary level. The second question is how the system behaves if the random variable ξ has a variation. Is the system stable? What is the corresponding stationary distribution? Does the long term average state deviate and how fast does it react on deviations from the equilibrium?

16.4 Discretization in the Value Iteration Approach

The behaviour of the system depends on the optimum quota rule $\pi(S,K)$ that solves (16.12) maximising future benefits. Focusing first on the deterministic case with discounting, we know [5] that $\pi(S,K)$ is an optimal rule, if there exists a value function $V(S,K)$ such that the Bellman equation

$$V(S,K) = \left(r(\pi(S,K),S,K) - c^i I(\pi(S,K),S,K) + \frac{1}{1+\alpha}V(S_{+1},K_{+1})\right), \tag{16.18}$$

is fulfilled, where S_{+1} and K_{+1} follow from the dynamic equations (16.1) and (16.2) and the behavior at the second level of the model, i.e. depending on values for $\pi(S,K), S$ and K. This optimality condition implies that if one can find the value function, one can also derive the optimal policy from

$$\pi(S,K) = \arg\max_{\pi}\left(r(\pi,S,K) - c^i I(\pi,S,K) + \frac{1}{1+\alpha}V(S_{+1},K_{+1})\right). \tag{16.19}$$

A value function and optimal policy can usually not be derived analytically. Therefore, we rely on a fixed point approach, where the function V is captured by a matrix with its values over a discretized state space. The optimum $\pi(S,K)$ and corresponding value function $V(S,K)$ are approximated by a value iteration approach. In this approach, one starts with an arbitrary valuation of function V and determines (16.18) iteratively, up to convergence for all state values (S,K). Practically, this works with a discretization of the state space (S,K) with vectors s,k, repeating (16.18) for each grid point as outlined in the algorithm.

We will first outline the approach for a deterministic setting and illustrate the results of a base case. After illustration, we extend to the consequences of having a random variable in the model.

Algorithm 1 :Pseudo code value iteration

Funct data, x,k vectors, ε; output π, Y matrices

1. $U = 0$ matrix
2. **for** all i, j
3. for state values $S = s_i, K = k_j$
4. $y_{ij} = \max_\pi \left(r(\pi,S,K) - c^i I(\pi,S,K) + \frac{V(S_{+1},K_{+1})}{1+\alpha} \right)$
5. **if** $\max_{ij}(y_{ij} - u_{ij}) - \min_{ij}(y_{ij} - u_{ij}) > \varepsilon$
6. $U = Y$ and go to step 2

16.4.1 Deterministic Elaboration

Using $S \in \{s_1, s_2, \ldots, s_i, \ldots, smax\}$ and $K \in \{k_1, k_2, \ldots, k_j, \ldots, kmax\}$ discretizes the state space, such that function $V(S,K)$ is approximated by the matrix Y with entrances $y_{ij} = V(s_i, k_j)$. For each matrix entrance (i, j), iteratively the minimum over π should be found of a function

$$y_{ij}(\pi) = r(\pi, x_i, k_j) - c^i I(\pi, x_i, k_j) + \frac{1}{1+\alpha} V(S_{+1}, K_{+1}), \tag{16.20}$$

where V is approximated by the matrix Y. This implies the use of interpolation of the value $V(S_{+1}, K_{+1})$ for the state (S_{+1}, K_{+1}), where the dynamics and decision lead to using the values in the matrix U. The implementation also requires considering the

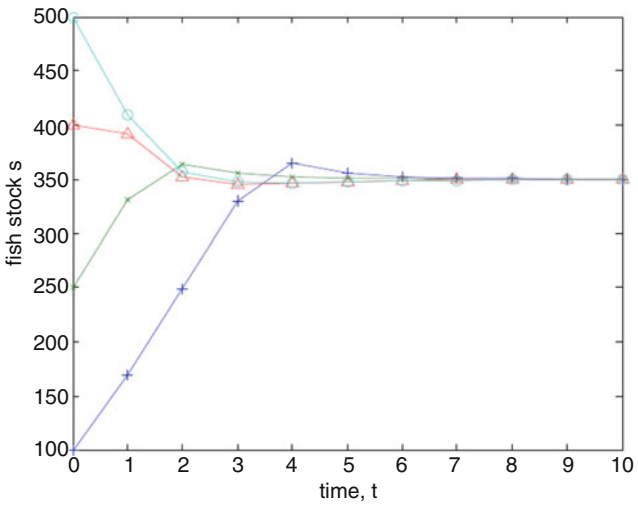


Fig. 16.1: Convergence of stock to equilibrium state, deterministic model; $K_0 = 9$

appropriate boundaries $s_1, smax$ and $k_1, kmax$ of the system and whether all combinations s_i, k_j are feasible. As discussed in Sect. 16.3.1, one does not have to consider fish stock values $S < \hat{s}$, as harvest is zero and quota can be taken $\pi = 0$. The upper bound $smax$ depends on the possibility to consider starting values $S_0 > m$. We can take $[s_1, xmax] = [\hat{s}, \max\{S_0, m\}]$. Following the reasoning in Sect. 16.3.1, it is appropriate to take $[k_1, kmax] = [0, \frac{1}{q}\ln(\frac{m}{\hat{s}})]$. The grid points s_i and k_j are not required to be equidistant in their corresponding range. A more refined grid, i.e. using more grid points, results into a better approximation of the value function V and the policy decision π.

The iterative minimisation of $y_{ij}(\pi)$ can be done by using a grid on a range $[0, qmax]$, or by using a one-dimensional minimisation algorithm.

Example 16.1. Our base case uses data taken from a study on North Sea plaice. This specimen is one of the main commercially exploited flatfish in the North Sea and is subject to increasing fishing pressure [4]: $m = 460, g = 0.74, q = 0.0139, d = 0.1, \alpha = 0.05, p = 1.83, c^i = 2.1, c^r = 0.25, c^e = 3.54$. For this base case, the level $\hat{s}$ under which fishing is not profitable, is $\hat{s} = 185.6$. The equilibrium value of fish stock predicted by (16.17) is $S = 349.5$.

The value iteration algorithm is run with $s_1 = 170, smax = 500, k_1 = 4, kmax = 70$ and taking 23 equidistant points s_i and k_j on each axis providing round numbers. The iterative minimisation of π is done by a standard one-dimensional minimisation algorithm FMINBND of MATLAB.

The iterative quota π converges after 20 iterations to the optimum one and the value function difference is converging, where difference between the minimum and maximum as defined in the algorithm reaches a value lower than $\varepsilon = 0.1$.

The behaviour of the system is sketched in Fig. 16.1 for four different starting values S_0 for the fish stock and an initial fishery capital stock of $K_0 = 9$. The system converges within 4 time steps to the equilibrium state. The stable dynamics of the fish stock due to (16.1) is helped by the fishing behaviour. For low values of fish stock, no fishing takes place and for values higher than the carrying capacity, harvesting reduces the stock due to low effort cost. The authority helps to reach the stable situation. For instance for $S = 200, K = 9$, the fishery sector would start harvesting approximately 14 tons. However, the authority keeps the quota at zero in order to promote the recovery of the fish stock, keeping long term welfare in mind.

16.4.2 Stochastic Implementation

Where ξ is a random variable, one should refine the Bellman equation. Expected value and probabilities come into play. The Bellmann equation (16.18) for the optimum solution $\pi(S, K)$ is extended towards

$$V(S,K) = \max_{\pi} \left(r(\pi, S, K) - c^i I(\pi, S, K) + \frac{1}{1+\alpha} E_\xi V(S_{+1}, K_{+1}) \right). \qquad (16.21)$$

Notice that the model requires to take the expected value E_ξ over future revenues and not over the current costs, because these costs only depend on the current state (S,K).

There are several practical ways to discretize the distribution of ξ. For instance, one can run over all possible grid points x_i, k_j for V_{+1} and assign a probability to these outcomes given decision π. This approach is quite cumbersome, as it requires re-calculating for many quota values and associated probability. The more common approach is to discretize the space of possible outcomes of the stochastic variable. A sharp way to do so is by using the quantiles of the lognormal distribution. This works by using an equidistant grid over the probability range $[0,1]$ with a step p_ξ and generating a discrete outcome space $\{\theta_1,\theta_2,\ldots,\theta_n\} = \{G^{-1}(p_\xi), G^{-1}(2p_\xi), G^{-1}(3p_\xi),\ldots,G^{-1}(1-p_\xi)\}$. The consequence of this operation is that the outcome space is truncated by the p_ξ-quantiles and every outcome has the same probability of occurrence. Actually, (16.21) is approximated by using

$$E_\xi V(S_{+1},K_{+1}) \approx p_\xi \sum_{i=1}^{n} V(S+\theta_i(gS\left(1-\frac{S}{m}\right)) - H(\pi,S,K),K_{+1}), \qquad (16.22)$$

where H is the harvest level chosen by the fishing sector at level 2 following from (16.8). Interpolation from matrix U is required in the state space to value V_{+1} for every possible outcome θ_i of the growth multiplier.

The ranges for the state variables do not change with respect to the deterministic model, as the possible outcomes θ_i are always positive. This means, that for $S < \hat{s}$, we have positive growth and for $S > M$, we have negative growth, so the same bounds can be used as in the deterministic case. In the stochastic model, positive growth can be much bigger than one. The calculation time for the value iteration increases, because each evaluation of a suggested quota π requires n values of the value function to be interpolated.

Example 16.2. In Example 16.1, the variable ξ is lognormal distributed with parameters $\sigma = 0.159$ and $\mu = -0.0126$. Distributing $n = 40$ points over the outcome space with $p_\xi = 0.025$ we have $\{\theta_1,\theta_2,\ldots,\theta_n\} = \{0.596, 0.716,\ldots,1.593\}$.

Running the value iteration algorithm, at each iteration using (16.22) for all grid points from $s \times k$ and interpolating U for the n outcomes θ_i, provides convergence of the optimum quota $\pi(S,K)$ after 15 iterations already.

16.4.3 Analysis of the Stochastic Model

The model can be characterized as a continuous space stationary MDP with its optimal policy $\pi(S,K)$ as depicted in Fig. 16.2. As illustrated, the discretisation to a finite number of states on the $s \times k$ grid does not directly provide a transition probability from state to state, as realisations will typically not find values on a grid point. Instead the optimal policy is derived by using interpolation of the value function.

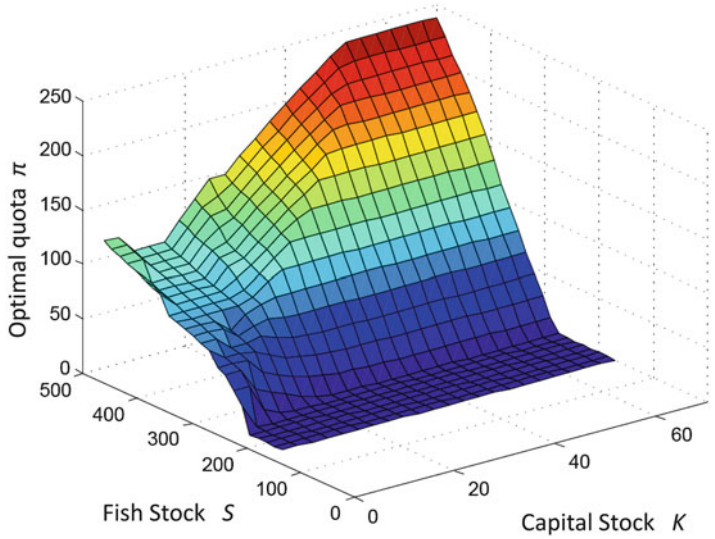


Fig. 16.2: Quota function found by value iteration stochastic model

We have observed for the deterministic case, that in principle the dynamic system convergences to an equilibrium in a short amount of time. To measure the behavior for the stochastic model the system has been simulated for 4 different starting values S_0 and 10 realisations of sample paths. These paths follow the optimum strategy using the values of the base case and the probability distribution of the example. The starting value for the capital stock is again taken as $K_0 = 9$. Figure 16.3 illustrates the fast convergence to absorption states around the equilibrium of the deterministic model.

16.5 Conclusions

In this chapter, a bi-level discounted Markov Decision Problem (MDP) model is analysed for setting optimal dynamic fishery policies. The model describes fish stock and economic dynamics. At the first level, an authority decides on the quota to be fished, keeping in mind long term social benefits. At the second level, fishermen react on the quota set and on the current state by deciding on their investments and fishery effort maximising the profit of that year.

The numerical solution by value iteration requires to limit the number of states. Further, span convergence goes very slowly if many states are included that are very unlikely to happen under an (nearly) optimal policy. The state space is discretized into a finite state space by studying the properties of a deterministic model. The

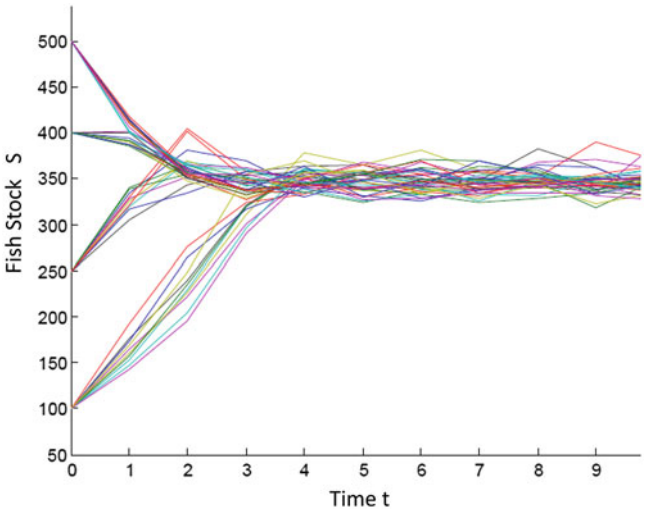


Fig. 16.3: Realisations of stochastic paths for four different starting values of fish stock following optimal quota setting; $K_0 = 9$

thus obtained ranges of the state space are applied to solve the stochastic problem on a uniform or equidistant grid. The action space is kept continuous, as well as the random growth in fish stock. Applying value iteration to the states on the grid, requires interpolation to estimate state values that are not on the grid.

Besides the ranges for the state space as obtained for an analyses of a deterministic problem, the choice of the grid points is arbitrarily set by the number of grid points that we allow. More grid points would make the solution more accurate, but would result in more states, and thus longer running times. Another way of increasing the accuracy, is applying Chebychev discretization, which assumes a more dense grid in areas of the state space with high probability of occurrence. Predetermining the areas of higher interest is however not straightforward, as estimates of the likelihood of each state are not available. Computing steady state probabilities by analysing a Markov chain would again first require discretization of the state space. Alternatively, such a probability distribution could be estimated by simulating an presumed optimal policy.

Given this analysis, an MDP approach based on value iteration is a feasible option to derive the optimal policy.

Acknowledgements This work has been funded by grants from the Spanish state (TIN2015-66680-C2-2-R) and, Junta de Andalucía (P11-TIC-7176), in part financed by the European Regional Development Fund (ERDF).

Appendix: Notation

This section describes the relation of the terminology in this chapter related to the general notation in the book.

S	Population size of fish measured in kton
K	Capital of fleet size state in horse power days
$\pi(S,K)$	Optimal policy with respect to quota level
$V(S,K)$	Value function
$r(\pi,S,K)$	One step revenue quota π in state (S,K)
$c^iI(\pi,S,K)$	One step investment cost for quota π in state (S,K)
U and Y	Iterate matrices of $V(S,K)$ over a discretized state space
α	Discount factor

References

1. M. Bailey, U.R. Sumaila, M. Lindroos, Application of game theory to fisheries over three decades. Fish. Res. **102**(1–2), 1–8 (2010)
2. J.R. Boyce, Optimal capital accumulation in a fishery: A nonlinear irreversible investment model. J. Environ. Econ. Manag. **28**(3), 324–339 (1995)
3. A.T. Charles, Optimal fisheries investment under uncertainty. Can. J. Fish. Aquat. Sci. **40**(12), 2080–2091 (1983)
4. L. Kell, P. Bromley, Implications for current management advice for north sea plaice (pleuronectes platessa l.): Part ii. increased biological realism in recruitment, growth, density-dependent sexual maturation and the impact of sexual dimorphism and fishery discards. J. Sea Res. **51**(3–4), 301–312 (2004)
5. M.L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (Wiley, New York, 1994)
6. G. Sethi, C. Costello, A. Fisher, M. Hanemann, L. Karp, Fishery management under multiple uncertainty. J. Environ. Econ. Manag. **50**(2), 300–318 (2005)
7. R. Singh, Q. Weninger, M. Doyle, Fisheries management with stock growth uncertainty and costly capital adjustment. J. Environ. Econ. Manag. **52**(2), 582–599 (2006)
8. M. Spence, Blue whales and applied control theory. Technical Report 108, Institute for Mathematical Studies, Stanford University (1973)
9. D. van Dijk, R. Haijema, E.M.T. Hendrix, R.A. Groeneveld, E.C. van Ierland, Fluctuating quota and management costs under multiannual adjustment of fish quota. Ecol. Model. **265**(0), 230–238 (2013)
10. D. van Dijk, E.M.T. Hendrix, R. Haijema, R.A. Groeneveld, E.C. van Ierland, On solving a bi-level stochastic dynamic programming model for analyzing fisheries policies: Fishermen behavior and optimal fish quota. Econ. Model. **272**, 68–75 (2014)

Chapter 17
Near-Optimal Switching Strategies for a Tandem Queue

Daphne van Leeuwen and Rudesindo Núñez-Queija

Abstract Motivated by various applications in logistics, road traffic and production management, we investigate two versions of a tandem queueing model in which the service rate of the first queue can be controlled. The objective is to keep the mean number of jobs in the second queue as low as possible, without compromising the total system delay (i.e. avoiding starvation of the second queue). The balance between these objectives is governed by a linear cost function of the queue lengths.
In the first model, the server in the first queue can be either switched on or off, depending on the queue lengths of both queues. This model has been studied extensively in the literature. Obtaining the optimal control is known to be computationally intensive and time consuming. We are particularly interested in the scenario that the first queue can operate at larger service speed than the second queue. This scenario has received less attention in literature. We propose an approximation using an efficient mathematical analysis of a near-optimal threshold policy based on a matrix-geometric solution of the stationary probabilities that enables us to compute the relevant stationary measures more efficiently and determine an optimal choice for the threshold value.
In some of our target applications, it is more realistic to see the first queue as a (controllable) batch-server system. We follow a similar approach as for the first model and obtain the structure of the optimal policy as well as an efficiently computable near-optimal threshold policy.
We illustrate the appropriateness of our approximations using simulations of both models.

D. van Leeuwen (✉) • R. Núñez-Queija
CWI, Amsterdam, The Netherlands
e-mail: daphne.vanleeuwen@gmail.com; Rudesindo.Nunez.Queija@cwi.nl

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_17

17.1 Introduction

We investigate a dynamic flow control problem arising in various applications. As a motivating illustration consider road traffic control, where trucks enter a crowded metropolitan area to supply goods in the city center. More often than not, such a scenario leads to clustering of transportation traffic near distribution facilities in the city. Our specific aim here would be to develop a control method that reduces long waiting lines of trucks at distribution centers located in or near cities. As a solution we investigate the effectiveness of a buffer location (i.e. a parking facility for trucks) near a distribution center to reduce the number of waiting trucks in busy areas, thereby giving more space to other traffic and reducing emissions.

Indeed, the buffer location will temporarily 'store' trucks and prevent overly crowded areas near the distribution center. On the down side, the introduction of the buffer location introduces an additional hop in the route for the trucks, creating potential inefficiency. When poorly operated, trucks may be waiting at the buffer location, while the service location at the distribution center may have cleared all the local backlog.

The problem setting described here illustrates a generic challenge in transportation logistics, manufacturing and production management. One may for example think of an asphalting machine that must be supplied with liquid asphalt at the correct pace, avoiding too long storage of the perishable material, but also maintaining sufficient supply to avoid expensive shutdown of the machine due to lack of material. In a production assembly line one can also imagine the necessity to balance the local inventory of assembly parts with the available space. And in road congestion management the traffic density near bottleneck junctions must be kept low enough to avoid traffic dead-lock, but on the other hand in the upstream direction the traffic flow should be sufficient to prevent unnecessary delay. We discuss and describe such control problems and design a generic strategy with practical applicability.

The above sketched situations are modeled by a controllable two-stage tandem queue. Referring to our initial motivation, the first stage represents the buffer and has infinite capacity. We first concentrate on the setting in which the server at the first stage can be controlled by an on-off switch. The second stage represents the distribution center for which we want to reduce the number of trucks. We seek an optimal trade-off between the reduction in the number of vehicles at the second stage and the additional delay caused by the on/off policy at the first stage. The optimal operation point is determined by the minimization of a cost function. This function accounts for waiting time costs in the buffering stage as well as costs for waiting at the distribution center. Arrivals to this system are modeled by a Poisson process and "service times" in both queues are exponentially distributed, which facilitates a formulation as a Markov Decision Problem (MDP).

This model has been studied extensively in literature and it is known that it is optimal to serve either at full speed or not to serve at all [14]. This type of strategies are referred to as "bang-bang strategies". Under certain cost assumptions it has been shown that the structure of the optimal service rate gives a switching strategy dividing the state space into two regions: one where the service rate is at its maximum

and one where service is paused. Such structural properties of optimality can be proven by showing the convexity of the value function in the Bellman optimality equation, see for example [14] and [6]. Similar convexity properties also hold for networks of queues [15]. Adopting the optimal control, several strategic questions can be answered as well, such as the desirable distance of the buffer location from the distribution center in order to regulate trucks optimally. In our model this distance is captured by the service rate at the first stage.

Full evaluation of the theoretically optimal strategy is often numerically infeasible, or at the least numerically overly time consuming for practical purposes. Several papers have looked at approximation techniques for this model. The fluid approximation developed in [1] works well when the second station works at a higher rate than the first station. We will summarize and complement that analysis with the opposite case (when the second station is the slowest) and show that then the optimal strategy can well be approximated by a threshold-based decision rule. The author in [10, pp. 439–441] developed a method to determine an approximation for the threshold value for the discounted reward MDP, which does not have a direct counterpart in the average reward problem. Alternatively, the authors in [5] propose a fluid approximation, focusing on a large deviations analysis. They determine the most probable evolution of the system toward rarely visited states, whereas we are concerned with controlling the system for optimal *average* behavior, i.e., we concentrate on states that are visited frequently.

In this chapter we discuss an approximation technique for the best threshold value in order to reduce the computational effort. This method uses matrix geometric analysis as described in [12]. Various papers, such as [4], have previously applied this method to speed up computation. The exposition in [7] for a tandem queue similar to ours is particularly relevant to develop our approximation as it gives an explanation of the blocks which are necessary to capture the details of our model.

In practice, service at the first station may not be limited to one job at a time. Indeed, in our primary example several trucks may jointly leave the parking facility if the waiting line at the distribution center is very short. Similarly, in manufacturing and production planning, several items may be produced or delivered at the same time. We therefore proceed to study an extension of the tandem queue with controllable service rate, allowing for batch service in the first station. A service batch corresponds, for example, to platoons of trucks jointly driving from the buffer location to the distribution center. In this setting, it is reasonable to maintain the service rate independent of the number of jobs that are jointly processed. We study the impact of batch service at the first server and determine the structure of the optimal policy. It turns out that the optimal queue level at the second queue is fully determined by the aggregate number of jobs in the two queues (i.e., the sum of the two individual queues). If the optimal levels can be determined, the optimal batch size is then easily computed for all states in the state space.

The batch model will be approximated in the same manner as the basic on-off model. The extension of the matrix geometric method for use in the batch model follows along the same lines as [11]. That reference focuses on a system which requires a minimum batch size to initiate service, and additionally, service can be

granted up to a predefined maximum batch size. In various other papers optimal batch sizes are determined via a trade-off between startup costs for service and costs per unit time for jobs in the system, see e.g. [3, 16]. Our model, however, does not have startup costs for batch service. We determine the optimal batch size leading to an optimal threshold level based on properties arising from the MDP formulation.

This chapter is structured as follows. We give a detailed description of the basic model in Sect. 17.2. In Sect. 17.3 we study the structure of the optimal policy for various choices of the parameters and observe that for most cases of interest in our context, the shape of the optimal switching curve suggests it can well be approximated by a threshold policy. In Sect. 17.4 we set out to determine the best choice for the threshold value using matrix-geometric analysis techniques. We then turn our attention to the batch-service model in Sect. 17.5 and follow the same program: we investigate the structural properties of the optimal strategy and develop a matrix-geometric representation to numerically determine a near-optimal strategy. In Sect. 17.6 we numerically study the appropriateness of the proposed strategies for both models. We conclude the chapter in Sect. 17.7.

17.2 Model Description: Single Service Model

Our model consists of two queues in tandem. As alluded to before, the second queue represents the actual service facility (distribution center, production plant or assembly line), whereas the first queue serves as a temporary buffer to alleviate congestion in the second queue. For analysis purposes we assume that jobs arrive to the first queue according to a Poisson process at rate λ and jobs *can be* processed at rate μ_1. After service in the first queue, jobs proceed to the second queue, for which the service rate is denoted with μ_2. For stability we assume both $\lambda < \mu_1$ and $\lambda < \mu_2$.

So as to control the number of jobs at the second station we introduce a binary decision at the first station, depicted in Fig. 17.1. The control mechanism may be interpreted as an on/off switch at the first station with two states: $\{0,1\}$. State 0 represents a service rate of 0, i.e. all jobs waiting at the first station will be blocked for service. State 1 represents the situation where each job at station 1 is served by rate μ_1 and continues to stage 2.

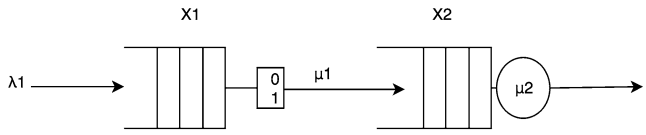


Fig. 17.1: The tandem queue with an on-off controlling mechanism

To formulate this as an optimization problem we introduce constant waiting costs c_{wait}, which are incurred *per job* and *per unit of time*. Jobs queueing at the second

station encounter additional costs indicated by c_{loc} which, in our introductory example, represents the costs of residing in the distribution area. Thus, the waiting cost at the first server is $c_1 = c_{wait}$ per job per unit of time, and at the second station it is $c_2 = c_{wait} + c_{loc}$. Naturally, we assume $0 < c_1 < c_2$. Due to larger costs at station 2, it is more advantageous to hold customers in queue 1 rather than in queue 2. However, one should avoid an empty station 2 when station 1 still has a backlog. We seek an efficient trade-off between these two effects.

We formulate the problem as a Markov Decision Process. The system will be observed at epochs of arrivals and service completions i.e. in discrete time. We use uniformization to discretize the Markov chain as described in Lippman [9]. Our discrete-time Markov Decision Process consists of the quadruple $\{\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{C}\}$. $\mathcal{S}$ represents the state space of the system, and is defined as $i = (x_1, x_2) \in \mathbb{N}^2$. Here x_i is the number of jobs at stations $i = 1$ and $i = 2$, respectively. $\mathcal{A}$ represents the action space; the set of actions that decision makers can take. In this problem $\mathcal{A} = \{0, 1\}$ representing either a blocked or an unblocked first server, respectively. Action 0 blocks service at station 1, i.e. no jobs can move from station 1 to station 2. For action 1 jobs are served at the first station at rate μ_1 and then move from station 1 to station 2. $\mathcal{P}$ contains the transition probabilities from state i to state j for action $a \in \mathcal{A}$; these can be written as $p^a(i, j)$. Finally, $\mathcal{C}$ denotes the cost function and will be written as $c^a(i)$ which is the expected cost per unit of time for each state $i = (x_1, x_2) \in \mathcal{S}$ and action $a \in \{0, 1\}$.

An optimal strategy satisfies Bellman's equation [2, 10]:

$$V^*(i) + g^* = \min_{a \in \mathcal{A}} \left\{ c^a(i) + \sum_{j \in \mathcal{S}} p^a(i, j) V^*(j) \right\} \text{ for } i \in \mathcal{S}, \tag{17.1}$$

where g^* and $V^*(i)$ give the optimal average reward and value function. The decision rule can be determined by:

$$f(i) = \operatorname{argmin}_{a \in \mathcal{A}} \left\{ c^a(i) + \sum_{j \in \mathcal{S}} p^a(i, j) V^*(j) \right\} \text{ for } i \in \mathcal{S}, \tag{17.2}$$

where $V^*(j)$ satisfies $V^*(i) + g^* = c^f(i) + \sum_{j \in \mathcal{S}} p^f(i, j) V^*(j)$. Note the slight abuse in notation in writing $c^f(i)$ and $p^f(i, j)$ instead of $c^{f(i)}(i)$ and $p^{f(i)}(i, j)$ as we should have according to our earlier notation. Our goal is to minimize the average cost and determine an optimal decision for each state.

To determine the optimal strategy in our tandem queue we use Eq. (17.2) where $c^a(i)$, with $i = (x_1, x_2)$, is given by $c_1 x_1 + c_2 x_2$. Recall that the cost c_1 consists only of the waiting cost per job at station 1 and c_2 is a combination of the waiting costs and additional costs for station 2, and that we assume $0 < c_1 < c_2$.

The transition probabilities $p^a(i, j)$ are determined by the transition rates in each state, applying uniformization as described in Lippman [9]. For action $a = 1$ (service in queue 1), we have for $x_1 \geq 0$ and $x_2 \geq 0$: $p^1((x_1, x_2), (x_1 + 1, x_2)) = \lambda / (\lambda + \mu_1 + \mu_2)$, $p^1((x_1 + 1, x_2), (x_1, x_2 + 1)) = \mu_1 / (\lambda + \mu_1 + \mu_2)$, and $p^1((x_1, x_2 + 1), (x_1, x_2) = \mu_2 / (\lambda + \mu_1 + \mu_2)$. On the boundary we have "dummy transitions"

leading to $p^1((0,x_2+1),(0,x_2+1)) = \mu_1/(\lambda+\mu_1+\mu_2)$, $p^1((x_1+1,0),(x_1+1,0)) = \mu_2/(\lambda+\mu_1+\mu_2)$, and $p^1((0,0),(0,0)) = (\mu_1+\mu_2)/(\lambda+\mu_1+\mu_2)$. Similarly, when $a=0$ (no service in queue 1), we have for $x_1 \geq 0$ and $x_2 \geq 0$: $p^0((x_1,x_2),(x_1+1,x_2)) = \lambda/(\lambda+\mu_1+\mu_2)$, and $p^0((x_1,x_2+1),(x_1,x_2) = \mu_2/(\lambda+\mu_1+\mu_2)$. Now there can be no service in queue 1, giving $p^0((x_1,x_2+1),(x_1,x_2+1)) = \mu_1/(\lambda+\mu_1+\mu_2)$. Finally, the remaining transitions on the boundary are $p^0((x_1,0),(x_1,0)) = (\mu_1+\mu_2)/(\lambda+\mu_1+\mu_2)$.

Successive Approximation (SA) will be used to calculate the optimal decision for each state so as to minimize average costs:

$$V_n(i) = \min_{a\in\mathcal{A}_i}\Big\{c^a(i) + \sum_{j\in\mathcal{S}} p^a(i,j)V^*_{n-1}(j)\Big\},$$

and

$$f_n(i) = \text{argmin}_{a\in\mathcal{A}_i}\Big\{c^a(i) + \sum_{j\in\mathcal{S}} p^a(i,j)V^*_{n-1}(j)\Big\}.$$

17.3 Structural Properties of an Optimal Switching Curve

We start our discussion of the optimal strategy with a numerical illustration for a particular example. Here and in the remainder of the chapter we will use $c_1 = 1$ and $c_2 = 3$ meaning that jobs incur waiting costs of 1 per unit of time and, only in queue 2, an additional location cost of 2 units. These values were also chosen in the example used by Meyn [10] for the discounted case. Our structural results hold for all values that satisfy $0 < c_1 < c_2$.

First we illustrate a dichotomy that occurs between the cases $\mu_1 < \mu_2$ i.e. the first server serves jobs at lower speed than the second server, and the opposite $\mu_1 > \mu_2$. The two graphs in Fig. 17.2 show the optimal strategy for the MDP under these two settings. A red color indicates that it is optimal to block service at the first stage. The green color implies a system working at maximum service speed at both stages. We observe that in both cases, the optimal action is prescribed by a so-called "switching curve" separating the green area from the red area. The shapes of the switching curves in the two graphs are a bit different. On the left, the curve eventually grows with a constant slope (this will be explained), whereas the graph on the right flattens for larger values of x_1. This difference appears to be fundamental to the two chosen parameter sets: one where the first server is slower than the second, and the opposite case. We will discuss this in more detail below.

To have a better understanding of the dynamics of the system operating under such a switching curve, we include the drift and trajectory diagrams displayed in Figs. 17.3 and 17.4. Irrespective of the shape of the switching curve, the drift above the curve is positive in the horizontal direction (due to arrivals at rate λ) and negative in the vertical direction (by departures from the second queue at rate μ_2). Note that because of the stability condition $\lambda < \mu_2$, the horizontal component of the drift is smaller than that in the vertical direction, but that is irrelevant for our

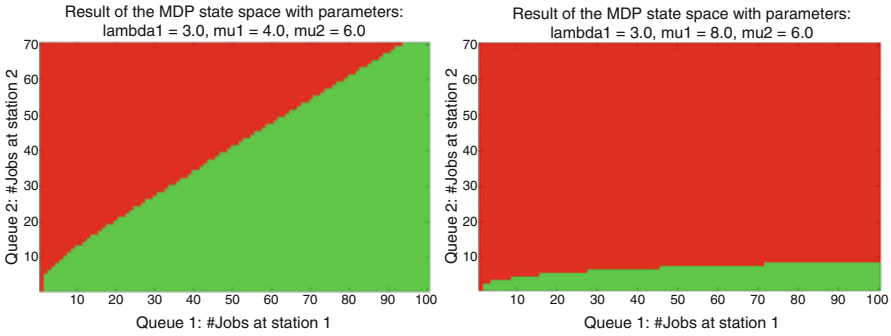


Fig. 17.2: An illustration of the optimal actions for all states. In the figures *red indicates* blocking (jobs are not served in queue 1) and *green indicates* that jobs at server 1 are served at rate μ_1

discussion here. Below the curve, the horizontal drift changes sign and has magnitude $\mu_1 - \lambda_1$, which is positive due to the stability condition $\lambda < \mu_1$. In the vertical direction the drift is $\mu_1 - \mu_2$. Here we observe a distinction between the case $\mu_1 < \mu_2$ in Fig. 17.3 and the case $\mu_1 > \mu_2$ in Fig. 17.4. In the first case ($\mu_1 < \mu_2$), we obtain a negative vertical drift and a corresponding direction toward the horizontal axis. If $\mu_1 > \mu_2$, the vertical drift is positive and the trajectory is directed toward the switching curve from both sides.

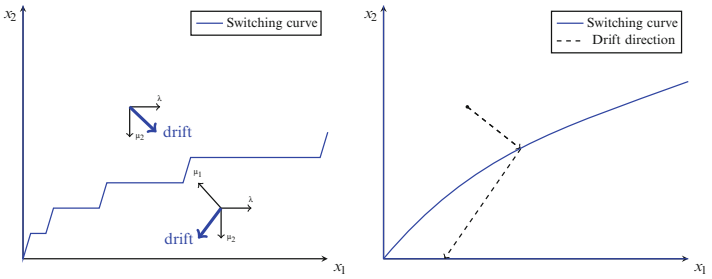


Fig. 17.3: An illustration of the drift above and below the switching curve (*left graph*) and a typical trajectory (*right graph*) ; $\mu_1 < \mu_2$

We now return to Fig. 17.2. The graph on the left suggests a close approximation of the switching curve by a linear function with positive intercept at the vertical axis. The graph on the right rather suggests an approximation by a horizontal line. The difference in behavior can be explained by the parameter choice. The linear increasing curve is the effect of a larger service capacity at the second stage, $\mu_1 < \mu_2$.

Intuitively, an optimal strategy must aim at avoiding an empty queue 2, when there are jobs in queue 1. The undesirable states are therefore located on the horizontal axis. If $\mu_1 < \mu_2$, the first queue can not "catch up" with the second queue, and therefore it should always provide sufficient inflow for queue 2 even at large system states. To further explain this fact, we refer to Fig. 17.3. The typical trajectory leads to the horizontal axis, which is the set of undesirable states. The linearly increasing switching curve avoids that the horizontal axis is hit at a very large level. When $\mu_1 > \mu_2$, the first server *can* catch up with the second server, because it serves at higher speed, thereby decreasing the probability of starvation of the second stage. All trajectories lead to the switching curve and then continue along the switching curve toward the origin. Hence, the size of the second queue can be maintained at a low level and consequently the switching curve flattens for larger levels of the first queue.. This fundamental difference leads to a likewise fundamentally different analysis of these two cases.

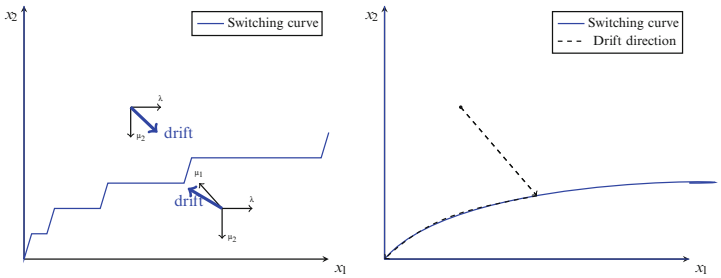


Fig. 17.4: An illustration of the drift above and below the switching curve (*left graph*) and a typical trajectory (*right graph*); $\mu_1 > \mu_2$

The first case, $\mu_1 < \mu_2$, has been investigated thoroughly in [1] using a fluid approximation. In this fluid approximation, the random trajectories are replaced by deterministic ones, characterized by their (expected) drifts. The fluid approximation can be shown to be the exact limit of the stochastic process under an appropriate scaling. The limiting fluid process can be shown to have an optimal linear switching curve, which translates into the optimal action for large system states, but lacks information about the optimal strategy near the origin. As we have observed, at the origin, the optimal switching curve for the stochastic model has a vertical offset. That offset can be approximated using perturbation methods [1], and turns out to give a good representation of the optimal strategy.

Unfortunately this method does not work when $\mu_1 > \mu_2$, which is the more relevant setting for many of our motivating applications. For example, the "buffer" location for the distribution center will likely not be located very far from the distribution center, which corresponds to relatively large values of μ_1. The above fluid approximation applied to $\mu_1 > \mu_2$ gives a switching curve that lies on the horizontal axis which suggests that the first server should never be operated. This is well explained by the sub-linear shape of the switching curve in the right graph of Fig. 17.2. On a linear scale, this graph vanishes for large system states.

We therefore set out to obtain an approximate analysis for the case $\mu_1 > \mu_2$. Theoretically it can be seen that the switching curve still increases indefinitely, albeit at a sub-linear pace. The flat shape however, implies that over large ranges of buffer levels in queue 1, the optimal action switches at a common buffer level of queue 2. This suggests that the optimal switching curve may well be approximated by a horizontal line, i.e. that a fixed threshold-based strategy should be close to optimal. Obtaining the optimal threshold value from the Bellman equations is computationally hard. In the sequel we use a matrix geometric analysis [12] to compute the best threshold value and compare it to the optimal strategy. An alternative method to approximate near-optimal threshold values for the discounted reward MDP was developed in [10, pp. 439–441]. Unfortunately, when applied to the average reward problem (under the usual limiting argument for discounted reward models [13, 8.2.2]) the threshold value becomes equal to infinity. For the purpose of this chapter our focus lies on an approximation for the average reward case. Alternatively, the authors of [5] approximate the curve using a large-deviations analysis. In that scaling, they obtain an asymptotically optimal switching level.

17.4 Matrix Geometric Method for Fixed Threshold Policies

We have argued that for the case $\mu_1 > \mu_2$ the optimal switching curve can perhaps be well approximated by a horizontal line. In order to compare the effectiveness of such fixed-threshold strategies, we set out to determine the relevant performance measures as functions of the threshold parameter K and then pick the best value of K. In this section we show that the resulting model falls into the class of Quasi-Birth-Death processes that allow for a matrix geometric solution. In order to cast our model into the framework of [12], we partition the state space into levels and phases, resulting in the generic structure of the generator matrix displayed in (17.3). In our model, each level will correspond with a fixed number of jobs in the first queue, and the phase within a level represents the number of jobs in the second queue. Thus, the generator matrix can be written in the block form of (17.3). Transitions between blocks correspond to a change in level (queue 1) and transitions within a block represent a change in phase (queue 2). The number of levels is therefore unbounded and the size of the block matrices (corresponding to the number of phases) is $K+1$, where K is the fixed threshold level.

Formally the state space can be described by $\mathcal{S} = \{(x_1, x_2) : x_1 \in \mathbb{Z}^+, 0 \leq x_2 \leq K+1\}$. The level index x_1 denotes the number of jobs at station 1 and x_2, the phase index, represents the number of jobs at station 2. The maximum number of jobs at station 2 is now bounded by the threshold K.

The generator matrix Q for this system is given by:

$$Q = \begin{bmatrix} B_0 & \Lambda & 0 & 0 & 0 & 0 & \cdots \\ M & D & \Lambda & 0 & 0 & 0 & \cdots \\ 0 & M & D & \Lambda & 0 & 0 & \cdots \\ 0 & 0 & M & D & \Lambda & 0 & \cdots \\ 0 & 0 & 0 & M & D & \Lambda & \cdots \\ 0 & 0 & 0 & 0 & M & D & \cdots \\ 0 & 0 & 0 & 0 & 0 & M & \cdots \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \end{bmatrix} \tag{17.3}$$

In this representation all blocks are square matrices of order $K+1$, and $M+D+\Lambda$ is a generator of a $K+1$ dimensional Markov process that follows the transitions of the second queue, *conditional on a non-empty first queue*. The stability condition is given by Neuts' mean drift criterion [12]: We define π to be the equilibrium distribution of a Markov process with generator $M+D+\Lambda$:

$$\underline{\pi}(M+D+\Lambda) = \underline{0} \text{ where } \pi\mathbf{e} = 1,$$

$\mathbf{e}$ being a $K+1$ dimensional vector with all entries equal to 1. The process with generator Q is stable if and only if $\underline{\pi}M\mathbf{e} > \underline{\pi}\Lambda\mathbf{e}$, i.e., the drift to higher levels should be strictly less than the drift to lower levels to guarantee stability of the system. For a fixed threshold level K the blocks are defined as follows:

$$B_0 = \begin{bmatrix} -\lambda & \cdots & \cdots & \cdots & 0 \\ \mu_2 & -a_1 & \cdots & \cdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \mu_2 & -a_1 & \vdots \\ 0 & \cdots & \cdots & \mu_2 & -a_1 \end{bmatrix}, \Lambda = \begin{bmatrix} \lambda & \cdots & \cdots & \cdots & 0 \\ \vdots & \lambda & \cdots & \cdots & \cdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \cdots & \cdots & \lambda & \vdots \\ 0 & \cdots & \cdots & \cdots & \lambda \end{bmatrix},$$

with $a_1 = \lambda + \mu_2$,

$$D = \begin{bmatrix} -a_2 & \cdots & \cdots & \cdots & 0 \\ \mu_2 & -a_2 & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \mu_2 & -a_2 & \cdots \\ 0 & \cdots & \cdots & \mu_2 & -a_1 \end{bmatrix}, M = \begin{bmatrix} 0 & \mu_1 & \cdots & \cdots & 0 \\ \vdots & \cdots & \mu_1 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \cdots & \cdots & \cdots & \mu_1 \\ 0 & \cdots & \cdots & \cdots & 0 \end{bmatrix},$$

with $a_2 = \lambda + \mu_1 + \mu_2$.

The repetitive block structure implies a matrix geometric form for the stationary probabilities corresponding to Q. Defining the $K+1$ dimensional vectors

$\underline{\pi_i} = (\pi_{i0} \dots \pi_{iK})$, where $\pi_{x_1 x_2}$ is the stationary probability of having x_1 jobs in the first and x_2 jobs in the second queue, the balance equations are:

$$\underline{\pi}_{i-1}\Lambda + \underline{\pi}_i D + \underline{\pi}_{i+1} M = \underline{0} \text{ for } i \geq 1,$$

and we can write:

$$\underline{\pi}_i = \underline{\pi}_{i-1} R \rightarrow \underline{\pi}_i = \underline{\pi}_0 R^i,$$

where the matrix R is the minimal non-negative solution to the following quadratic matrix equation:

$$R^2 M + RD + \Lambda = R.$$

Via iteration we may determine R (there are alternative and more efficient routines, see [8]). Once we have determined a solution for R we can include the boundary conditions. It then remains to compute $\underline{\pi}_0$ via the remaining boundary equation:

$$\underline{\pi}_0 B_0 + \underline{\pi}_1 \Lambda = 0.$$

For a unique solution we impose the normalization condition that the probabilities sum to 1. This gives:

$$\underline{\pi}_0 \mathbf{e} + \sum_{i=1}^{\infty} \underline{\pi}_0 R^i \mathbf{e} = 1 \text{ or equivalently } \underline{\pi}_0 (I - R)^{-1} \mathbf{e} = 1.$$

In order to compute the cost function we determine the average queue length for both queues:

$$E[x_1] = \pi_0 R (I - R)^{-2} \mathbf{e},$$
$$E[x_2] = \pi_0 (I - R)^{-1} J,$$

where J is the column vector $(0, 1, \dots, K-1)^T$.

To determine the optimal threshold we simply minimize costs over all thresholds:

$$\min_{K \in N} \{\pi_0 (I - R)^{-1} * (c_1 R (I - R)^{-1} \mathbf{e} + c_2 J).\} \tag{17.4}$$

Now we can compute the best threshold level with respect to the costs and compare it with the MDP policy. From now on we will refer to this policy as the "optimal threshold" policy, not implying that this policy is overall optimal, but rather that it is optimal among the threshold policies. Determining the optimal threshold is much less computationally demanding than finding the optimal strategy using the MDP approach.

17.5 Model Description: Batch Transition Model

We now extend the model allowing for batch services in the first queue. As we elaborated on in the introduction, processing multiple jobs at the first server to prevent starvation at the second server is a logical choice for various applications of this model. To clarify in what ways this model extends the previous one, we will describe it while referring to the details of the first model. To gain understanding of the new model a graphical representation is shown in Fig. 17.5. Compared to Fig. 17.1 we can see that the first queue is serving N jobs in one single service instead of handling jobs individually.

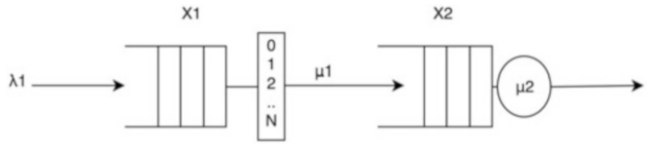


Fig. 17.5: Graphical representation of the tandem queue with batch service in the first queue

So as to allow the first queue to serve in batches we extend the decision space from $a \in \{0,1\}$ to $a \in \{0,\ldots,x_1\}$ when the number of jobs in the first queue is x_1 (the set of possible actions is thus dependent on the current state). The value of a corresponds to the chosen batch size, which is naturally limited by the number of jobs in the first queue. Next, we adapt the transition probabilities described in Sect. 17.2. For $i = (0,0)$ we have

$$p^a(i,j) = \begin{cases} \frac{\lambda}{\lambda+\mu_1+\mu_2} & \text{if } j = (1,0) \\ \frac{\mu_1+\mu_2}{\lambda+\mu_1+\mu_2} & \text{if } j = (0,0) \end{cases},$$

and for $i = (x_1,x_2) \neq (0,0)$:

$$p^a(i,j) = \begin{cases} \frac{\lambda}{\lambda+\mu_1+\mu_2} & \text{if } j = (x_1+1,x_2) \\ \frac{\mu_1}{\lambda+\mu_1+\mu_2} & \text{if } j = (x_1-a,x_2+a) \text{ for } a \in \{0,\ldots,x_1\}. \\ \frac{\mu_2}{\lambda+\mu_1+\mu_2} & \text{if } j = (x_1,x_2-1) \qquad \text{or } i = j = (x_1,0) \end{cases} \tag{17.5}$$

For this system there's always a strategy that stabilizes it as long as $\lambda < \mu_2$, irrespective of the value of $\mu_1 > 0$. This is obvious, since we can always choose to serve all jobs in queue 1 in a single batch, no mater how many there are. Different from the single service model, there will be no clear distinction between the cases $\mu_1 > \mu_2$ and $\mu_1 < \mu_2$, because the first station is always able to 'catch up' with the second station, even for $\mu_1 < \mu_2$. In the batch transition model we will see that the switching curve always flattens for larger x_1 values as is illustrated in Fig. 17.6. More

details on this figure will be given when we investigate the structural properties of the batch service model.

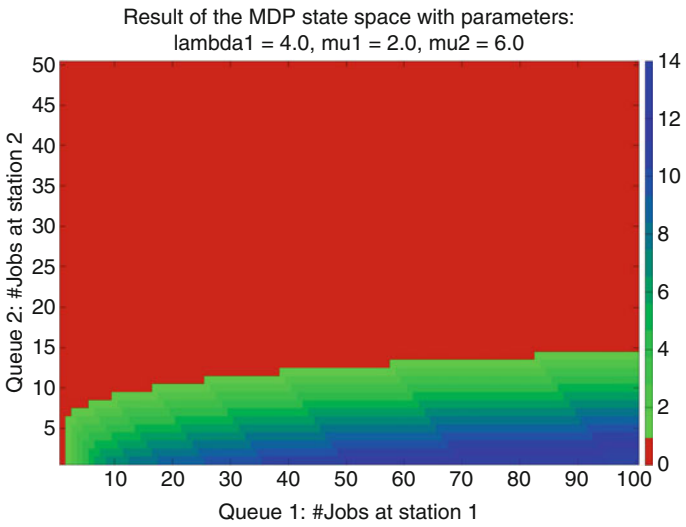


Fig. 17.6: Output of the MDP for the batch service model for parameter set $\lambda = 4$, $\mu_1 = 2$ and $\mu_2 = 6$. Each *color* represents a different optimal batch size for the current state

17.5.1 Structural Properties of the Batch Service Model

We will apply similar numerical calculations to show the main structural properties of the batch service model and compare with the single service model. For the single service model, the state space was divided into two regions depending on the optimal decision. For the batch transition model similar shapes are encountered when grouping states with the same optimal decision. In Fig. 17.6 states colored in red correspond as before to blocking of service at the first queue. The next layer corresponds to states in which the optimal batch size is one, then we have states with an optimal batch size of two, etcetera. This figure suggests that the optimal trajectories of the process are near the curve dividing red from green colored states. In this numerically obtained graph, the shape of this curve is again sublinear; we will indeed show that this is confirmed when investigating a scaled version of the process. We note once more, that this shape is not restricted to particular parameter settings, as was the case in the single batch model where the sublinear shape corresponded to the choice $\mu_1 > \mu_2$. The larger jumps now allow the system to move to the switching curve in one step from any state.

Although this is rather difficult to see from Fig. 17.6, we observe that the optimal strategy can again be characterized by the *single* switching curve separating the red states from all others. Given the total number of jobs in the system, say $x_1 + x_2 = N$, the optimal action is to serve a jobs in the first queue such that $(x_1 - a, x_2 + a)$, which also has N jobs in total, is *on* the switching curve. Should this value of a be negative (this happens when (x_1, x_2) is in the red area), then no jobs should be served in the first queue.

A graphical representation of the optimal transitions can be seen in Fig. 17.7 for two different values of the total number of jobs in the system: $N = N'$ and $N = N''$. All states on a diagonal $x_1 + x_2 = N$ "point" to the same destination state at the intersection of this line and the switching curve.

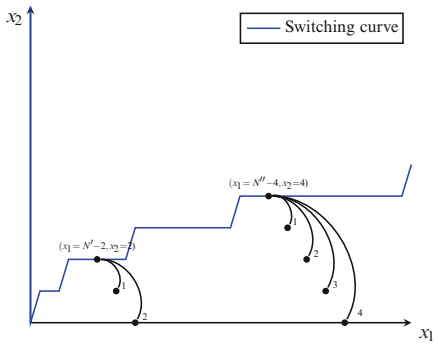


Fig. 17.7: A graphical representation of the optimal batch size for two examples

17.5.2 Matrix Geometric Method with Batch Services

Similar to the case $\mu_1 > \mu_2$ in the single service model, we wish to approximate the (sublinear) switching strategy with a horizontal line, thereby implementing a threshold-based strategy with, say, threshold value K. This model falls into the class of G/M/1 type Markov chains that admit a matrix geometric solution for the stationary distribution. Note that, for a fixed threshold value, the condition $\lambda < \mu_2$ is not sufficient for stability. For sure, the system can not be stable if $\lambda \geq K\mu_1$, because $K\mu_1$ is the maximum rate at which jobs can be pushed from the first station. The additional condition $\lambda < K\mu_1$ is therefore necessary, but certainly not sufficient either. The precise stability condition can be shown to be

$$\lambda < \mu_1 \left(K \left(\frac{\mu_2}{\mu_1 + \mu_2} \right)^K + \sum_{k=1}^{K-1} k \frac{\mu_1}{\mu_1 + \mu_2} \left(\frac{\mu_2}{\mu_1 + \mu_2} \right)^k \right). \tag{17.6}$$

This can be obtained by interpreting the right hand side of this inequality as the exact departure rate from the first station if that station were saturated (i.e., starting with infinitely many jobs in station 1). We will not make this precise here, as we can obtain this equation from Neuts' mean drift condition, which we will do below.

To define the batch transition model in matrix geometric form extra blocks must be added into the generator matrix, that allow for the larger transition jumps. As was described in the previous section, the batch size can be derived from the switching curve, effectively redistributing the total number of jobs over the two queues (with the obvious limitation that no jobs can be moved from the second to the first queue).

The generator matrix Q_{batch} now has the following structure:

$$Q_{batch} = \begin{bmatrix} B_0 & \Lambda & 0 & 0 & 0 & \cdots & \cdots \\ B_1 & D & \Lambda & 0 & 0 & \cdots & \cdots \\ B_2 & M_1 & D & \Lambda & 0 & \cdots & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \cdots \\ B_{K-1} & M_{K-2} & M_{K-3} & \cdots & \cdots & \ddots & \cdots \\ B_K & M_{K-1} & M_{K-2} & \cdots & \cdots & \ddots & \cdots \\ 0 & M_K & M_{K-1} & \ddots & \ddots & \cdots & \cdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \end{bmatrix} \tag{17.7}$$

The 0th level of the process represents the boundary states, comparable to the single service model. The matrices B_0, Λ and D are defined as before. Because of the batch services, the block matrices below the diagonal must be adapted. The matrices B_k, $k-1,2,\ldots,K$ correspond to transitions for which the first queue is emptied. This is only possible when there are 1 up to K jobs in the first queue, and the second queue has sufficient space left to accommodate the batch size:

$$B_1 = M = \begin{bmatrix} 0 & \mu_1 & 0 & \cdots & 0 \\ 0 & 0 & \mu_1 & 0 & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & \mu_1 \\ 0 & \cdots & \cdots & \cdots & 0 \end{bmatrix}, B_2 = \begin{bmatrix} 0 & 0 & \mu_1 & \cdots & 0 \\ 0 & 0 & 0 & \mu_1 & \vdots \\ \vdots & \ddots & \ddots & 0 & \mu_1 \\ \vdots & \cdots & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 \end{bmatrix}, \cdots, B_K = \begin{bmatrix} 0 & \cdots & \cdots & 0 & \mu_1 \\ 0 & \cdots & \cdots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \cdots & \cdots & \cdots & \vdots \\ 0 & \cdots & \cdots & \cdots & 0 \end{bmatrix},$$

The transitions corresponding to batch services that do not lead to an empty first station are grouped in the matrices M_k:

$$M_K = B_K, M_{K-1} = \begin{bmatrix} 0 & \cdots & \cdots & 0 \\ 0 & \cdots & 0 & \mu_1 \\ 0 & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & 0 \end{bmatrix}, \cdots, M_1 = \begin{bmatrix} 0 & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \mu_1 \\ 0 & \cdots & \cdots & 0 \end{bmatrix}$$

From Neuts' mean drift criterion [12] we can obtain the stability criterion in (17.6). Similar to the single service model we now define π to be the equilibrium distribution of a Markov process with generator $\Lambda + D + \sum_{k=1}^{K} M_k$:

$$\underline{\pi}(\Lambda + D + \sum_{k=1}^{K} M_k) = \underline{0} \text{ where } \underline{\pi}\mathbf{e} = \underline{1}.$$

The process with generator Q is stable if and only if the drift condition $\underline{\pi} \sum_{k=1}^{K} M_k \mathbf{e} > \underline{\pi}\Lambda\mathbf{e}$ is satisfied.

Again, following [12], the stationary distribution has a matrix-geometric structure $\underline{\pi}_i = \underline{\pi}_0 R^i$, for $i = 1, 2, \ldots$, where the matrix R is the minimal non-negative solution of

$$\Lambda + RD + \sum_{k=1}^{K} R^{k+1} \cdot M_k = 0.$$

The boundary equations now read

$$\underline{\pi}_0 \sum_{k=0}^{K} R^k \cdot B_k = \underline{0},$$

and the normalization condition is

$$\underline{\pi}_0 \sum_{k=0}^{\infty} R^k \cdot \mathbf{e} = \underline{\pi}_0 \cdot (I - R)^{-1} \cdot \mathbf{e} = 1.$$

Again, by computing the stationary distributions for varying values of the threshold K, we may determine the best value of the threshold in terms of the average cost as we did for the single service model using (17.4). Finally we can compare this result with the optimal MDP policy.

17.6 Simulation Experiments

In this section we illustrate the effectiveness of the threshold policies, obtained using the matrix geometric representation, with the optimal strategies from the MDP formulation. For our comparison, we will simulate both classes of strategies, although for the threshold strategies the reported results can also be directly obtained after computing the stationary distribution.

In Fig. 17.8 the costs and the average queue lengths are plotted for varying service rate at station 1. We observe that the performance of the best threshold policy is almost identical to that of the optimal MDP policy. The right hand graph also shows that the two policies are very close to each other in terms of the average queue lengths. The discontinuities in the curves corresponding to the threshold policies correspond to parameter choices where the optimal threshold value shifts by one. As can be expected, the discontinuities for the MDP policies are much less pronounced, as the optimal switching curve may shift only for a small number of states.

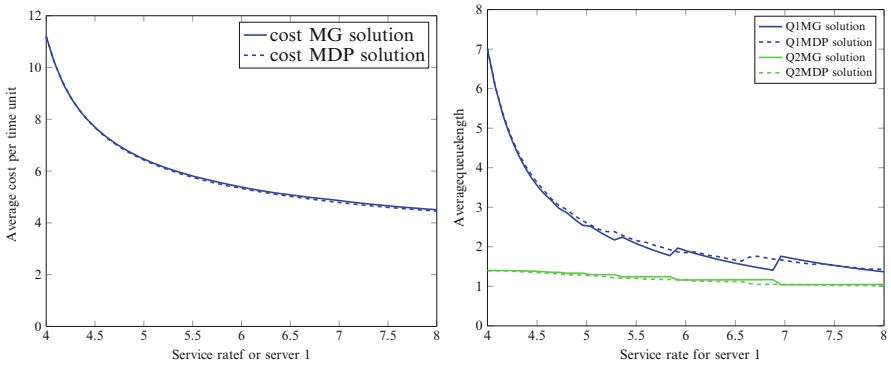


Fig. 17.8: Comparison of the optimal MDP policy model and the threshold-based approximation for the single service model for varying service rate μ_1. The parameters are: $\lambda = 3, \mu_1 \in [4,8], \mu_2 = 6$

For the batch service model, the simulation results of the MDP and the threshold strategies are reported in Fig. 17.9. We may now take μ_1 to be smaller than λ without compromising the stability of the optimal policy as long as condition (17.6) is satisfied.

We again observe a remarkable fit in terms of cost, for almost all values of the service rate at the first station. As could be expected, the costs are lower compared to the single service model. As for the queue lengths, we observe that the batch service mode allows to keep the first station at lower levels, but the queue lengths at the second station remain at roughly the same level. For now we defer further comparisons between the models with and without batch services and focus on comparisons between the MDP and the threshold strategies. Due to the more aggressive service mode, the costs of the threshold policies are much less smooth than in Fig. 17.8 and the optimal mean queue lengths oscillate more for larger values of μ_1. Indeed, changing the threshold value by one has a much larger impact on the resulting policy (that aims to bring the queue length back to the horizontal switching curve in a single service run). The strong fluctuations in queue lengths, make it all the more surprising that the costs of the best threshold policy remain close to those of the optimal MDP strategy.

We have now compared the rightfulness of the approximating threshold strategies. Next we compare the gain of having batch service in the first station. In Fig. 17.10 the best threshold values are determined, again for increasing service rate at station 1. The single service model is not stable for $\lambda \geq \mu_1$. We observe that the threshold strategies only perform badly in the single service model when the system approaches instability. For a large range of values with $\mu_1 < \mu_2$, the single-service threshold strategy performs almost as well as the batch-service threshold strategy, although in that case the optimal switching curve for the single-service model has a rather steep (linear) ascent. It is quite surprising that the costs are comparable for the two models, as long as μ_1 does not approach the stability limit (i.e., remains $> \lambda$).

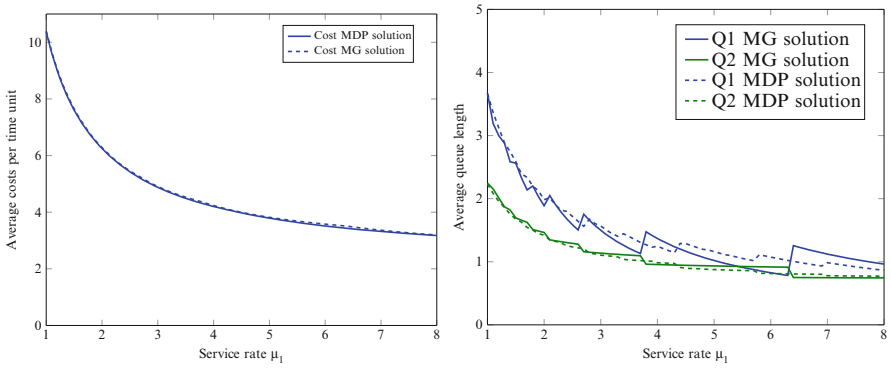


Fig. 17.9: Comparison of the optimal MDP strategy and the threshold-based approximation for the batch service model for varying service rate μ_1. The parameters are: $\lambda = 3, \mu_1 \in [1,8], \mu_2 = 6$

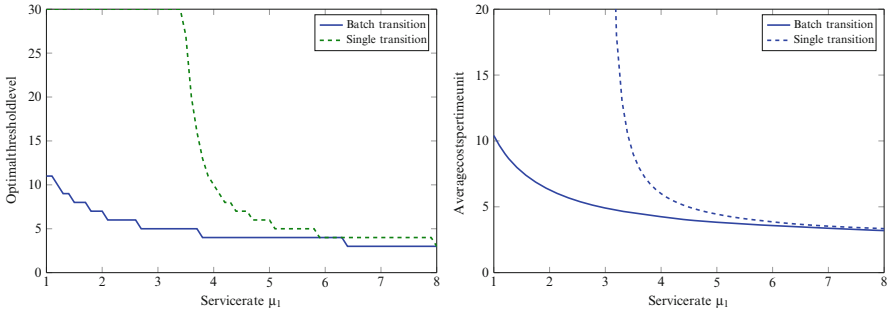


Fig. 17.10: Comparison of optimal threshold strategies for the single service and batch service model; $\lambda = 3, \mu_1 \in [1,8], \mu_2 = 6$

The optimal threshold levels do, naturally, differ: that of the single service model is considerably larger than in the batch service model (as could be expected).

Table 17.1 displays the costs for the two models for various threshold levels. The batch transition model performs better for all threshold levels, also non-optimal levels, but the difference is not very pronounced. The main advantage of the batch-service model is that the costs are not that sensitive to the exact value of the threshold. For the single-service model the costs are much more sensitive and small errors in the threshold value may lead to considerable loss of efficiency.

17.7 Conclusion

We have investigated the optimal control of the number of jobs in an expensive service station, by regulating the flow from a preceding buffer station. We started by determining the optimal control policy using a Markov Decision Problem formulation.

	$\lambda = 4, \mu_1 = 5, \mu_2 = 6$		$\lambda = 4, \mu_1 = 7, \mu_2 = 6$	
K	Single service	Batch transition	Single service	Batch transition
3	–	18.39	13.04	7.34
4	50.17	7.16	7.20	6.00
5	14.01	6.87	6.75	6.19
6	11.23	7.06	6.78	6.47
7	10.42	7.29	6.90	6.70
8	10.12	7.47	7.00	6.87
9	10.00	7.61	7.10	6.99
10	9.96	7.71	7.17	7.07
11	9.95	7.78	7.22	7.12
12	9.96	7.83	7.25	7.16

Table 17.1: *Comparing the costs of single and batch services for various threshold levels.*

The optimal strategy can in general be characterized by a switching curve; the shape of this curve is determined by whether or not the first station has a larger service rate than the second. If so, then the optimal switching curve is rather flat, otherwise it increases approximately linearly. When the optimal switching curve is flat, it can well be approximated by a horizontal one, which corresponds to a fixed threshold strategy. Besides their practical relevance due to the simplicity of implementation, threshold-based strategies have the advantage that they allow a much more detailed analysis. By casting a threshold based control into the framework of Markov models with a matrix geometric stationary distribution, we can efficiently compute the best threshold level. For this "optimal" threshold level, we indeed verified that it performs very closely to the optimal MDP strategy.

For some of our motivating examples, the "feeding" process from the buffer station need not necessarily be by individual jobs only. It is quite natural to allow multiple jobs to be served in a single service run from the first station. For this model we again formulated and studied the corresponding MDP and established that the optimal switching curve always has a flat shape, irrespective of the speeds of the servers. Again, threshold based strategies were shown to be much more efficiently solvable and have close to optimal performance.

Surprisingly, the best single-service threshold and the best batch-service threshold policies were found to give comparable performance, unless the single-service threshold policies were near their stability limit (the arrival rate being near the service rate of the first station). In the latter case, batch-service threshold strategies profit from their larger stability region.

Our results also translate into practical design rules. First of all, the simplicity of threshold based rules make them much easier to implement in practical scenarios. Note that the optimal switching curve policy requires to operate a different threshold level depending on the load in the first station.

A further insight is that for single service mode at the first station, it is important that the service rate in that station is large enough (preferably larger than, or at least comparable to that at the second station). When applied to the distribution center setting, this implies that the parking facility should not be too far from the distribution center; in fact, the travel time between the two should be smaller than, or comparable to, the unloading time at the distribution center. If multiple jobs can be simultaneously transferred between the two stations, the distance is not a major issue. In that case, performance is rather insensitive to the service speed in the first station (unless that speed is very low).

References

1. F. Avram, *Optimal Control of Fluid Limits of Queueing Networks and Stochasticity Corrections.* Lectures in Applied Mathematics, vol. 33 (Springer, New York, 1997), pp. 1–36
2. R.E. Bellman, *Dynamic Programming.* (Princeton University Press, Princeton, NJ, 2003) Republished 2003: Dover
3. G.L. Curry, R.M. Feldman, An M/M/1 queue with a general bulk service rule. Nav. Res. Logist. Q. **32**(4), 595–603 (1985)
4. A. El-Rayes, M. Kwiatkowska, G. Norman, Solving infinite stochastic process algebra models through matrix-geometric methods. School of Computer Science Research Reports, University of Birmingham (1999)
5. A. Gajrat, A. Hordijk, A. Ridder, Large-deviations analysis of the fluid approximation for a controllable tandem queue. Ann. Appl. Probab. **13**, 1423–1448 (2003)
6. G. Koole, Convexity in tandem queues. Probab. Eng. Inf. Sci. **18**(1), 13–31 (2004)
7. G. Latouche, M. Neuts, Efficient algorithmic solutions to exponential tandem queues with blocking. SIAM J. Algebr. Discr. Meth. **1**, 93–106 (1980)
8. G. Latouche, V. Ramaswami, A logarithmic reduction algorithm for quasi-birth-and-death processes. J. Appl. Probab. **30**, 650–674 (1993)
9. S.A. Lippman, Applying a new device in the optimisation of exponential queueing systems. Oper. Res. **23**(4), 687–710 (1975)
10. S.P. Meyn, *Control Techniques for Complex Networks* (Cambridge University Press, Cambridge, 2008). ISBN 978-0-521-88441-9 hardback
11. M. Neuts, A general class of bulk queues with Poisson input. Ann. Math. Stat. **38**(3), 759–770 (1967)
12. M. Neuts, *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach*. (Courier Dover Publications, Mineola, 1981)
13. M.L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (Wiley, New York, 1994)

14. Z. Rosberg, P.P. Varaiya, J. Walrand, Optimal control of service in tandem queues. IEEE Trans. Autom. Control **27**(3), 600-610 (1982)
15. R.R. Weber, S. Stidham, Optimal control of service rates in networks of queues. Adv. Appl. Probab. **19**, 202–218 (1987)
16. J. Weiss, The computation of optimal control limits for a queue with batch services. Manag. Sci. **25**(4), 320–328 (1979)

Part V
Communications

Chapter 18
Wireless Channel Selection with Restless Bandits

Julia Kuhn and Yoni Nazarathy

Abstract Wireless devices are often able to communicate on several alternative channels; for example, cellular phones may use several frequency bands and are equipped with base-station communication capability together with WiFi and Bluetooth communication. Automatic decision support systems in such devices need to decide which channels to use at any given time so as to maximize the long-run average throughput. A good decision policy needs to take into account that, due to cost, energy, technical, or performance constraints, the state of a channel is only sensed when it is selected for transmission. Therefore, the greedy strategy of always exploiting those channels assumed to yield the currently highest transmission rate is not necessarily optimal with respect to long-run average throughput. Rather, it may be favourable to give some priority to the exploration of channels of uncertain quality.

In this chapter we model such on-line control problems as a special type of Restless Multi-Armed Bandit (RMAB) problem in a partially observable Markov decision process framework. We refer to such models as Reward-Observing Restless Multi-Armed Bandit (RORMAB) problems. These types of optimal control problems were previously considered in the literature in the context of: (i) the Gilbert-Elliot (GE) channels (where channels are modelled as a two state Markov chain), and (ii) Gaussian autoregressive (AR) channels of order 1. A virtue of this chapter is that we unify the presentation of both types of models under the umbrella of our newly defined RORMAB. Further, since RORMAB is a special type of RMAB we also present

J. Kuhn (✉)
The University of Queensland, St. Lucia, Brisbane, Australia

University of Amsterdam, Amsterdam, The Netherlands
e-mail: j.kuhn@uva.nl

Y. Nazarathy
The University of Queensland, St Lucia, Brisbane, Australia
e-mail: y.nazarathy@uq.edu.au

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_18

an account of RMAB problems together with a pedagogical development of the Whittle index which provides an approximately optimal control method. Numerical examples are provided.

18.1 Introduction

Communication devices are often configured to transmit on several alternative channels, which may differ in their type (e.g. WiFi versus cellular) or in their physical frequencies. Further, due to physical transmitter limitations, a device can only use a limited number of channels at any given time. Thus, the question arises which channels to select for transmission so as to maximize the throughput that is achieved over time.

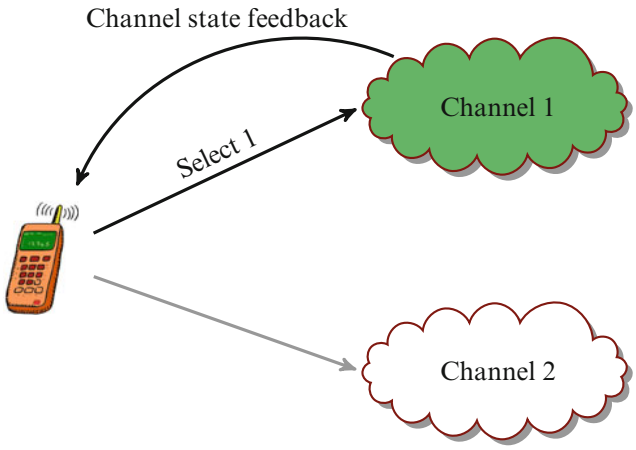


Fig. 18.1: A transmitting device needs to choose whether to transmit on Channel 1 or Channel 2. Transmitting on a channel results in immediate channel state feedback

To illustrate the problem, we consider the scenario depicted in Fig. 18.1. At every discrete time instance, the transmitter has the choice to use either channel 1 or channel 2. The channels cannot be used in parallel, for instance, due to hardware limitations and/or energy constraints. The selected channel then yields an immediate reward that depends on the condition of the channel (e.g. the reward may be measured as the number of bits successfully transmitted). Consequently, an observation of the state of that channel is also obtained. The unselected channel on the other hand is not observed in this time instance.

Ideally, the transmitter would choose channels in a way that achieves the largest throughput over time. However, the nature of communication channels is often stochastic and thus the transmitter does not know the current state of each channel with certainty. A good prediction of the channel state can be obtained when there is strong dependence between the current state of a channel and its state in the (recent)

past. Such *channel memory* can for example be caused by other users interfering on the same channel, multipath of physical signals, or other persistent disturbances.

In utilizing channel memory, to make wise channel selection decisions, the transmitter needs to balance a trade-off between exploitation and exploration: On the one hand, based on the controller's belief regarding the current state of each channel, it may make sense to choose the channel expected to transmit the highest number of bits over the next time slot. On the other hand, it may be sensible to check the condition of the other channel so as to decrease uncertainty regarding its current state. How should channels be selected, based on the information available, so as to maximize the long-run expected throughput?

A first step towards answering this control problem is to devise suitable *channel models*; that is, models that capture channel behaviour reasonably well and at the same time are simple enough to be mathematically tractable. To capture such a dependence, channel states are often modelled as Markov processes. One such very simple process is the so-called *Gilbert-Elliot channel* (GE), where there are two possible states, 0 ("bad") and 1 ("good"), and transitions between states occur in a Markovian manner. The application of the GE model in channel selection and specifically opportunistic spectrum access is motivated by its ability to capture bursty traffic of primary users [7]. Due to its simplicity it has been very popular in modelling channel selection problems; refer to the literature review in Sect. 18.5.

Another class of models, which has only recently come to attention in the context of wireless channel selection [4, 20], are *Gaussian autoregressive processes of order* 1 (which we denote by AR). Here, the channel state is a continuous random variable following a normal distribution, and its evolution is determined by a simple linear recursion perturbed by Gaussian noise. It has been found that Gaussian autoregressions model the logarithmic signal-to-noise ratio of a channel reasonably well; for details see [1].

The virtue of both the GE and the AR model is that they are quite tractable, yet allow to capture the exploration–exploitation trade-off that the controller faces. The models are simple in the sense that the *belief* which the controller maintains about the state of the channel is neatly summarized by sufficient statistics. In the GE case, this sufficient statistic is given by the conditional probability of being in the good state, given the information that is available to the controller at the time. In the AR case, it is sufficient to keep track of the conditional mean and variance of the state, which quantify the expected gain from exploitation and the need for exploration, respectively.

An optimal policy for the channel selection problem needs to balance this exploration–exploitation trade-off. While such a policy could in principle be computed by dynamic programming, this is typically computationally infeasible in practice [33]. However, recognizing that the problem is essentially a Restless Multi-Armed Bandit (RMAB) problem, we may apply techniques from RMAB theory to find a (near-)optimal solution: the well-known Whittle index [41] (a generalization of the celebrated Gittins index [8, 10]).

Our main focus in this chapter is on the RMAB formulation of the problem. We call this special type of RMAB the *Reward-Observing Restless Multi-Armed*

Bandit (RORMAB) problem. While the chapter does not contain new results, it is unique in that it provides a unified treatment of both the GE and the AR approaches for channel models, and considers also the channel selection problem in the mixed case where some of the channels are modelled as GE while the others are AR. This is of interest in networks where some but not all of the channels may be subject to user interference.

The remainder of this chapter is structured as follows. In Sect. 18.2 we formulate the RORMAB problem, and present the GE and AR models in a unified manner. In Sect. 18.3 we motivate the use of index policies and in particular the Whittle index. The presentation can be used as a stand-alone brief account of RMAB problems. In Sect. 18.4 we show how to use channel models to evaluate the Whittle index numerically, and use it as a solution strategy for an example channel selection problem. In the latter section we also provide a number of performance comparisons. Section 18.5 contains a literature survey, and points out some open problems.

18.2 Reward-Observing Restless Multi-Armed Bandits

In this section we formulate the RORMAB problem within the context of wireless channel selection. This type of problem is a special case of a Partially Observable Markov Decision Process (POMDP) as considered in [36]. An MDP is partially observable if the decision maker does not know the current state of the system with certainty. In our setting, where we consider a network of d channels, the partially observable state of the system can be represented as d-dimensional, and corresponds to the joint state information of the individual channels.

We consider channels $X_1(t),\ldots,X_d(t)$, operating as independent Markov processes in discrete time $t \in \mathbb{N}_0$. We assume that the models and their parameters are known but do not have to be the same for each channel. At every time instance, the decision maker chooses a subset $a(t)$ of k channels, $a(t) \subset \{1,\ldots,d\}$. For every selected channel $i \in a(t)$ a reward $R^i\big(X_i(t)\big)$ is obtained and the value of $X_i(t)$ is observed, where each $R^i(\cdot)$ is assumed to be a known, deterministic function from the underlying state space to $\mathbb{R}$. The other channels $i \notin a(t)$ are not observed and do not yield a reward.

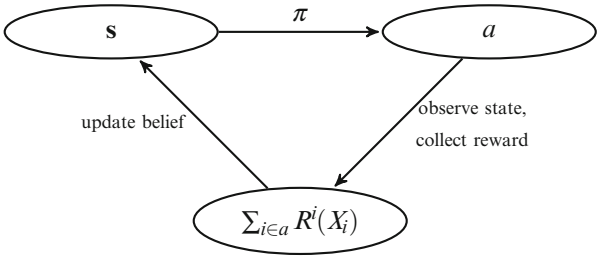
In an ideal situation, at every decision time the controller would choose those channels that yield the highest reward $\sum_{i\in a(t)} R^i\big(X_i(t)\big)$. Unfortunately, this cannot generally be achieved because channel states are stochastic and the values of $X_1(t),\ldots,X_d(t)$ are not known at decision time t.

However, because the channel states are sequentially dependent due to the channel memory, the controller can use information about the previous state of a channel to make predictions about the current state. The accuracy of the prediction depends on the *age* of the information, i.e. the number of time steps ago that a channel was last observed. This number is denoted by $\eta_i(t) := \min\{\tau \geq 1 : i \in a(t-\tau)\}$, so that the last time channel i was chosen is given by $t-\eta_i(t)$. The information available to the transmitter at time t (prior to making its decision) can then be summarized and represented by $Y(t) := \big(Y_1(t),\ldots,Y_d(t)\big)$, where for $i = 1,\ldots,d$,

$$Y_i(t) = \Big(\eta_i(t), X_i\big(t-\eta_i(t)\big)\Big).$$

Based on $Y_i(t)$, the controller's *belief* about the state of channel i at time t is summarized by $F_i(x) := \mathbb{P}\big(X_i(t) \le x \mid Y_i(t)\big)$, the conditional distribution of channel i given the information collected up to that time. For the channel models we consider, this probability distribution is characterised by scalar- or vector-valued sufficient statistics. That is, for channel i there exists a parameter $s_i(t)$ that fully specifies the probability distribution of $X_i(t)$ given the information $Y_i(t)$. In our first model (GE as described below), $F_i(\cdot)$ is a Bernoulli distribution, so $s_i(t)$ is the "success probability". In our second model (AR as described below), $F_i(\cdot)$ is a normal distribution, hence, $s_i(t)$ is a two-dimensional vector specifying the conditional mean and conditional variance. Using the terminology common in literature on POMDP, we refer to $s_i(t)$ as the *belief state* of channel i at time t; indeed $s_i(t)$ represents our belief concerning the state of the channel.

In summary, as time evolves from t to $t+1$, given the current belief state $\mathbf{s} := (s_1, \ldots, s_d)$ and a channel selection policy π, the following chain of actions takes place:



Objective. Our aim is to find a policy π so as to maximize the accumulated rewards over an infinite time horizon as evaluated by the *average expected reward criterion*

$$G^\pi(\mathbf{s}) := \liminf_{T\to\infty} \frac{1}{T} \mathbb{E}_{\mathbf{s}}^\pi \left[\sum_{t=0}^{T-1} \sum_{i\in a(t)} R^i\big(X_i(t)\big) \right], \tag{18.1}$$

where the subscript indicates conditioning on $X_i(0)$ being distributed with parameter s_i. Note that other reward criteria have been considered, including finite horizon problems and/or discounted rewards/costs [34]. In this chapter we focus on the average reward criterion (18.1) in order not to overload the exposition.

It can be proven formally [5, 36] that a POMDP with partially observable states $X_i(t)$ and rewards $R^i\big(X_i(t)\big)$ is equivalent to a fully observable MDP with states $s_i(t)$ and rewards $r^i\big(s_i(t)\big) := \mathbb{E}_{s_i(t)}\big[R^i\big(X_i(t)\big)\big]$ in the sense that the best throughput that can be achieved is the same for both, and it is achieved by the same (optimal) policy. This justifies that we consider the MDP with states $s_i(t)$ in the remainder of this chapter.

Belief State Evolution. For RORMAB, the decisions determined by a policy π affect the updating of the belief state based on the *observation update* mapping $\mathcal{O}_i(\cdot)$ and the *belief propagation* operator $\mathcal{T}_i(\cdot)$ as follows:

$$s_i(t+1) = \begin{cases} \mathcal{O}_i\big(X_i(t)\big), & \text{if } i \in a(t), \\ \mathcal{T}_i\big(s_i(t)\big), & \text{if } i \notin a(t). \end{cases} \tag{18.2}$$

The *observation update* mapping $\mathcal{O}_i(\cdot)$ determines how the belief state of channel i is updated when that channel is selected for transmission. In this case we observe $X_i(t)$, and hence its realization can be used by the observation update rule when implementing the controller. Further, for analytical, modelling and simulation purposes, when $i \in a(t)$, the distribution of $X_i(t)$ is determined by the known value $s_i(t)$, so $X_i(t)$ can be replaced by a generic random variable coming from this distribution.

The *belief propagation* operator $\mathcal{T}_i(\cdot)$ defines the update of the belief state of a channel when it is not selected for transmission. Because in this case no new observation is obtained, the update is deterministic.

Since a channel may remain unobserved for several consecutive time slots, it is useful to also consider $\mathcal{T}_i^k(\cdot)$ (the k-step operator obtained by applying $\mathcal{T}_i(\cdot)$ k times) as well as attracting fixed points of the operator $\mathcal{T}_i(\cdot)$. As we describe below, in both the GE and the AR model, the k-step operator has an explicit form and converges to a unique attracting fixed point; this is useful for understanding the dynamics of the model.

We now specify the observation update and belief propagation operations in the context of each of the two channel models.

Gilbert-Elliot (GE) Channels. In this case $X_i(t)$ is a two state Markov chain on the state space $\{0,1\}$, where 0 represents a "bad" state and 1 is a "good" state of the channel. The transition matrix can be parametrized as

$$P_i = \begin{bmatrix} p_i^{00} & p_i^{01} \\ p_i^{10} & p_i^{11} \end{bmatrix} = \begin{bmatrix} 1-\gamma_i\overline{\rho_i} & \gamma_i\overline{\rho_i} \\ \overline{\gamma_i}\overline{\rho_i} & 1-\overline{\gamma_i}\overline{\rho_i} \end{bmatrix},$$

where we denote $\bar{x} := 1-x$. One standard parametrization of this Markov chain uses transition probabilities $p_i^{01}, p_i^{10} \in [0,1]$ (and sets $p_i^{00} = \overline{p_i^{01}}$, $p_i^{11} = \overline{p_i^{10}}$). Alternatively we may specify the stationary probability of being in state 1, denoted by $\gamma_i \in [0,1]$, together with the second eigenvalue of P_i, denoted by $\rho_i \in \big[1-\min(\gamma_i^{-1}, \overline{\gamma_i}^{-1}),\ 1\big]$. Then ρ_i quantifies the time-dependence of the chain. If $\rho_i = 0$, the chain is i.i.d., otherwise there is memory. Specifically, when $\rho_i > 0$ there is positive correlation between successive channel states and when $\rho_i < 0$ that correlation is negative. The relationship between these parametrisations is given by $\gamma_i = p_i^{01}/(p_i^{01}+p_i^{10})$, and $\rho_i = 1-p_i^{01}-p_i^{10}$. The parametrization with transition probabilities $p_i^{\ell k} \in [0,1]$ is standard. As opposed to that, our alternative parametrization in terms of γ_i and ρ_i has not been used much in the literature. Nevertheless, we find it captures the behaviour of the model in a useful manner, especially when carrying out numerical comparisons.

As the Bernoulli distribution is fully specified by the success probability, it suffices to keep track of this parameter. That is, we have

$$s_i(t) = \omega_i(t) := \mathbb{P}\big(X_i(t) = 1 \,|\, Y_i(t)\big),$$

and hence the belief state space of channel i, denoted by S_i, is given by the interval $[0,1]$. Now the *observation update* operation is defined by:

$$\mathcal{O}_i(x) = \begin{cases} p_i^{01}, & \text{if } x = 0\,, \\ p_i^{11}, & \text{if } x = 1\,. \end{cases}$$

That is, if the observed channel was "bad" ($x = 0$), then the chance of a good channel is given by the entry p_i^{01}, and otherwise ($x = 1$) by p_i^{11}. The *belief propagation* operation is

$$\mathcal{T}_i(\omega) = \omega p_i^{11} + \overline{\omega} p_i^{01} = \rho_i\, \omega + \gamma_i\, \overline{\rho_i}\,.$$

This follows by evaluating the probability of $\{X_i(t+1) = 1\}$ based on $\omega_i(t)$ and the probability transition matrix P_i. It is a standard exercise for two state Markov chains (recurrence relations) to show that the k-step transition probability, and thus the k-step belief propagation operator, takes the form

$$\mathcal{T}_i^k(\omega) = \gamma_i + \rho_i^k(\omega - \gamma_i).$$

Note that γ_i is a fixed point of this operator, and the sequence $\mathcal{T}_i^k(\omega)$ converges to this fixed point. Further note that if $\rho_i > 0$, this sequence is monotonic whereas if $\rho_i < 0$, it oscillates about γ_i as it converges to it. The case of $\rho_i = 0$ is not of interest in terms of decision making because in that case there is no channel memory.

Gaussian Autoregressive (AR) Channels. In this case the channel states follow an AR process of order 1, that is,

$$X_i(t) = \varphi_i X_i(t-1) + \varepsilon_i(t)\,,$$

with $\big\{\varepsilon_i(t) : t \in \mathbb{N}_0\big\}$ denoting an i.i.d. sequence of $\mathcal{N}\big(0, \sigma_i^2\big)$ random variables. We assume $|\varphi_i| < 1$, in which case the processes are stable in the sense that as time evolves they converge to a stationary version. Note that if $\varphi_i \in (0,1)$, the states are positively correlated over time; for $\varphi_i \in (-1,0)$ the correlation is negative. The case $\varphi_i = 0$ may be neglected as it corresponds to observations being independent. Linear combinations of Gaussian random variables are still Gaussian, and hence, their conditional distribution at time t is fully described by the conditional mean $\mu_i(t)$ and the conditional variance $\nu_i(t)$. That is, the sufficient statistic (vector) for the state of channel i is:

$$s_i(t) = \big(\mu_i(t),\, \nu_i(t)\big)\,.$$

In this AR case, the *observation update* operation is:

$$\mathcal{O}_i(x) = \big(\varphi_i x,\, \sigma_i^2\big)\,.$$

This is due to the fact that an observation of x at time t implies a predicted expected value of $\varphi_i x$ at time $t+1$ with variance σ_i^2. In contrast, the *belief propagation* operation is given by

$$\mathcal{T}_i(\mu_i, \nu_i) = \left(\varphi_i \mu_i, \varphi_i^2 \nu_i + \sigma_i^2\right). \tag{18.3}$$

It is easy to show by recursion of the mean and the variance that the k-step belief propagation is:

$$\mathcal{T}_i^k(\mu_i, \nu_i) = \left(\varphi_i^k \mu_i, \varphi_i^{2k} \nu_i + \frac{1-\varphi_i^{2k}}{1-\varphi_i^2} \sigma_i^2\right). \tag{18.4}$$

The belief state space in this case is $S_i = \mathbb{R} \times [\nu_i^{\min}, \nu_i^{\max})$ where $\nu_i^{\min} = \sigma_i^2$ and $\nu_i^{\max} = \sigma_i^2/(1-\varphi_i^2)$. The attracting fixed point of $\mathcal{T}_i(\cdot)$ is the mean-variance pair $(0, \nu_i^{\max})$. It is further interesting to note that the second coordinate of the belief state can only attain values in a countable subset of $[\nu_i^{\min}, \nu_i^{\max})$. This is because when the channel is selected, the conditional variance decreases to the value $\nu_i^{\min}$, and thus, ν_i in (18.4) is always proportional to σ_i^2, where the factor is given by a geometric series in φ_i^2. Observe further that, since $\nu_i < \nu_i^{\max}$ and because $|\varphi_i| < 1$, it always holds that the variance increases when updated with $\mathcal{T}_i(\cdot)$, that is, the decision maker's uncertainty regarding the state of the channel indeed grows as long as no new observation is obtained.

Mixed Model Example. Having specified the GE and AR channel models, we now consider a mixed model example, which is also used for numerical illustration in Sect. 18.4. Research in this field to date seems to have focussed on problems with channels of the same type (mostly GE, some AR); it is therefore interesting to investigate a mixed channel model example, where a proportion $q \in [0,1]$ of the channels is GE and the others are AR. This can occur in examples where the dominating phenomena of some of the channels is user interference (GE channels), while for other channels the key feature is slow-fading behaviour (AR channels).

Our model parameters are α_i, γ_i for $i = 1, \ldots, qd$ (GE channels) and φ_j, σ_j^2 for $j = qd+1, \ldots, d$ (AR channels); we assume that qd is an integer.

For the purpose of exposition we consider the following stylised case of reward functions:

$$R^i(x_i) = \frac{x_i - \gamma_i}{\sqrt{\gamma_i(1-\gamma_i)}}, \quad \text{and} \quad R^j(x_j) = \frac{\sqrt{1-\varphi_j^2}}{\sigma_j} x_j, \tag{18.5}$$

where x_i is a value observed in GE channel i and x_j is the value observed in AR channel j. We specifically choose these functions so that the steady state values of rewards from both channels have zero-mean and unit-variance, hence making the channels equivalent in these terms. That is, in the case where the controller does not have additional state information, the controller obtains the same mean and variance on any channel chosen.

The state space of the MDP with (joint belief) states $\mathbf{s} = (s_1, \ldots, s_d)$, with scalars s_i, $i = 1, \ldots, qd$, and 2-dimensional vectors s_j, $j = qd+1, \ldots, d$, is given by:

$$S := [0,1]^{qd} \times \prod_{j=qd+1}^{d} \mathbb{R} \times [v_j^{\min}, v_j^{\max}).$$

An optimal policy π for such an MDP is usually not available in closed form. It can then be computed approximately with the aid of dynamic programming algorithms, on a discretized and truncated state space. This is feasible with sufficient accuracy only if d is very small (and indeed this is carried out as part of the numerical examples provided in Sect. 18.4 for $d = 2$). With more channels, the computational task quickly becomes intractable. Therefore, we resort to a sensible index heuristic (the Whittle index), which we present in the next section.

18.3 Index Policies and the Whittle Index

In this section we explain the idea behind the use of index policies and specifically the Whittle index, a generalization of the well-studied Gittins index [8, 41]. Whittle proposed this type of index as a heuristic solution to RMAB problems. We first describe a general form of RMAB problem so as to put our specific RORMAB problem in context.

Consider state processes $s_1(t), \ldots, s_d(t)$ subject to an action $a(t) \subset \{1, \ldots, d\}$ which selects k of the d processes at each time. In the (more general) context of RMAB, we refer to each of these processes as an *arm* of a bandit. Based on the control decisions captured in the control set $a(\cdot)$, each of the processes evolves either according to an *active* mapping $\mathcal{A}_i(\cdot)$ if $i \in a(t)$, or according to a *passive* mapping $\mathcal{P}_i(\cdot)$ otherwise. This can be represented as follows:

$$s_i(t+1) = \begin{cases} \mathcal{A}_i\big(s_i(t), U_i(t)\big), & \text{if} \quad i \in a(t), \\ \mathcal{P}_i\big(s_i(t), U_i(t)\big), & \text{if} \quad i \notin a(t). \end{cases} \tag{18.6}$$

Here, $\{U_i(t), i = 1, \ldots, d\}$ are independent i.i.d. (driving) sequences of uniform $(0,1)$ random variables. An alternative representation is to use Markovian transition kernels, one for the active operation and one for the passive operation.

Remark: Comparing (18.6) and (18.2) it is evident that our RORMAB channel selection problem is a special case of the RMAB problem. Channels and belief states of the RORMAB correspond to arms and states of the RMAB, respectively. In the RORMAB, the active mapping $\mathcal{A}_i(s,u)$ is replaced by $\mathcal{O}_i\big(F^{-1}(u;s)\big)$ where $F^{-1}(\cdot\,;s)$ is the inverse probability transform generating a random value of the state, distributed with parameter(s) s; and the passive mapping $\mathcal{P}_i(s,u)$ does not depend on the random component and is replaced by $\mathcal{T}_i(s)$. In the remainder of this section

we depart from the RORMAB context and present a brief exposition of RMAB and the Whittle index.

The RMAB problem arose as a generalisation of the Multi-Armed Bandit (MAB) problem. For the MAB it is assumed that $\mathcal{P}_i(s,u) = s$; that is, unselected arms do not evolve. This problem is known to be solved optimally in great generality by the celebrated Gittins index [8]. Whittle's RMAB generalization allows for "restless" state processes, where arms keep evolving also while they are not used for transmission (although not necessarily according to the same transition kernel). This modelling framework is more realistic for the channel selection problem but also appears in a variety of other application areas; see [41] for further examples. For RMAB problems, index policies are typically not optimal. However, the Whittle index, which we present below, has in many cases proven to be asymptotically optimal with respect to the average reward criterion as the number of arms grows large [38, 40].

We now first describe index policies in general before we turn to Whittle's optimization problem and the associated Whittle index policy.

Index Policies. Index policies are defined in terms of functions $\iota_1, \ldots, \iota_d$ such that ι_i maps the current state of arm i to a certain priority index, irrespective of the current state of any other arm.

Definition 18.1. Let $\mathbf{s} := (s_1, \ldots, s_d)$ denote the vector of states in a system with d arms. An index policy π_ι with stationary decision rule δ_ι, activates those k arms that correspond to the k largest indices,

$$\delta_\iota(\mathbf{s}) = \underset{a \subset \{1,\ldots,d\}: |a|=k}{\arg\max} \sum_{i \in a} \iota_i(s_i) .$$

Ties are broken arbitrarily, but in compliance with the requirement that k arms have to be selected.

For an intuitive justification as to why *index policies* may work well in large systems, consider the following: Pick an arbitrary arm and suppose we want to decide whether to select it as active or not (passive), based on the current state. Generally, we would make our decision dependent on the states of the remaining arms. In this way, our decision strategy is highly influenced by the proportion of arms that are in a certain state. In a large system with many arms, however, this proportion can usually be expected to remain relatively stable over time. In this sense, the larger the system, the less important it is for us to consider other arms; we always find ourselves in roughly the same situation for decision making. In conclusion, in a system with many arms, little is lost if we make decisions for each arm solely based on its current state, disregarding the current state of any other arm in the system.

How to best define the index functions ι_i? A simple example is the myopic index. In the context of RORMAB, where states are actually belief states, it is defined by $\iota_i^M(s_i) = \mathbb{E}_{s_i}\left[R^i(X_i)\right]$. Thus, under the myopic policy the transmitter greedily chooses those channels that promise the largest immediate rewards ("exploitation").

However, as one may expect, it turns out that such a policy is not necessarily optimal (see our numerical examples in Sect. 18.4, as well as the literature survey in Sect. 18.5). It may be favourable to give some priority to "exploring" other channels in order to decrease the transmitter's uncertainty with respect to their current state.

Moving back to the more general RMAB, this motivates us to consider the more sophisticated Whittle index, which takes the possible need for considering future states (or "exploration" in the case of RORMAB) into account. To derive his heuristic, Whittle relaxed the constraint that exactly k arms have to be selected at each time point, and replaced it by the requirement that k arms are selected *on average*. Since the latter constraint is weaker, the optimal throughput (value/gain of the MDP) under this constraint is an upper bound for the optimal throughput that can be achieved in the original problem. We shall see that this relaxation allows to formulate the decision making problem as a Lagrange optimization problem, from which Whittle obtained a rule for determining $\imath_i(s_i)$.

Whittle's Optimization Problem. For ease of exposition we consider an MDP with a finite state space $S = S_1 \times \cdots \times S_d$. We further remain in the setting where arms that are not selected do not yield a reward (this assumption is easily generalized so that the Whittle index is applicable much more broadly [41]). Under suitable regularity conditions, the optimal long-run average throughput rate is independent of the initial state of the system (see e.g. [5]); in this section we assume that we are in such a setting.

Recall the definition of the average reward criterion, Eq. (18.1). For a time horizon T ($T \to \infty$) we sum up the rewards that are obtained from the selected arms at each time point. Equivalently, we could group selected arms according to their states, and keep track of how many arms were selected while being in a specific state, over the whole time horizon. That is, rather than considering each time step t separately and adding up rewards as obtained at each time step, we can consider how many arms were in a certain state when selected, and multiply this proportion with the reward that is obtained from an arm in that state. If we do so for all states, then the total is equivalent to the value of the average reward as $T \to \infty$.

This is the viewpoint we are taking in this section; it is inspired by the exposition in [27]. Define $p_i(s)$ as the expected long-run fraction of time that arm i is selected when it currently is in state $s \in S_i$; that is,

$$p_i(s) := \lim_{T\to\infty} \frac{1}{T} \mathbb{E}^{\pi} \left[\sum_{t=0}^{T} \mathbb{1}\{ i \in a(t), s_i(t) = s \} \right].$$

Subject to Whittle's relaxation, we can then formulate the optimization problem as the Linear Programming (LP) problem:

$$G^W = \max_{p} \sum_{i=1}^{d} \sum_{s\in S_i} r^i(s)\, p_i(s), \quad \text{subject to} \quad \sum_{i=1}^{d} \sum_{s\in S_i} p_i(s) = k, \tag{18.7}$$

where $r^i(s)$ denotes the reward that is obtained from selecting arm i when its state is s (as before $r^i(\cdot)$ is a known, deterministic function). Formulating this as an equivalent Lagrangian optimization problem we obtain:

$$\begin{aligned}\mathcal{L}(\lambda) &= \max_p \sum_{i=1}^{d} \sum_{s\in S_i} r^i(s)\, p_i(s) - \lambda \left(\sum_{i=1}^{d} \sum_{s\in S_i} p_i(s) - k \right) \\ &= \max_p \sum_{i=1}^{d} \sum_{s\in S_i} \big(r^i(s) - \lambda\big) p_i(s) + \lambda k \\ &= \sum_{i=1}^{d} \mathcal{L}_i(\lambda) + \lambda k,\end{aligned} \tag{18.8}$$

with

$$\mathcal{L}_i(\lambda) = \max_{p_i} \sum_{s\in S_i} \big(r^i(s) - \lambda\big) p_i(s)\,. \tag{18.9}$$

Since in (18.8) there is no longer a common constraint for the arm, each arm can be optimized separately through (18.9). By strong LP duality we know that there exists a Lagrange multiplier λ^* that yields $\mathcal{L}(\lambda^*) = G^W$. LP complementary slackness ensures that (assuming $\lambda^* \neq 0$) any optimal solution to (18.8) must satisfy the relaxed constraint, and is therefore also optimal for Whittle's relaxed problem (18.7). It was observed by Whittle [41] that we can interpret the Lagrange multiplier as a cost for selecting an arm (or equivalently, as a subsidy for not selecting an arm). That is, imposing a cost of λ^* on the selection of an arm causes the controller to select k arms on average under a policy that optimises G^W.

Indexability and the Whittle Index. In accordance with [41], we make the following reasonable regularity assumption.

Assumption: Arm i is *indexable*, that is, the set of states for which it is optimal to select arm i decreases monotonically from S_i to $\emptyset$ as the cost λ increases from $-\infty$ to ∞. This property holds for every arm in the system.

While this assumption is intuitively appealing, it turns out that it does not generally hold [38, 41], and proving its validity can be surprisingly difficult [8]. In the context of RORMAB, it has been verified for the GE model as considered in [22], and numerical evidence suggests that it also holds for the AR model [20]; the latter, however, is still to be proven.

Indexability implies that for each arm i there exists a function of the current state, $\lambda_i(s)$, such that it is optimal to select the arm whenever $\lambda_i(s) > \lambda$ and to leave it passive otherwise (the decision maker is indifferent when $\lambda = \lambda_i(s)$). In this sense, $\lambda_i(s)$ measures the "value" of arm i when it is in state s. Furthermore, applying this policy to all arms in the case where $\lambda = \lambda^*$ (that is, selecting arm i whenever $\lambda_i(s) > \lambda^*$) results in a policy that is optimal for Whittle's relaxed problem (18.7).[1] This motivates choosing the index function $\iota_i(\cdot)$ as $\iota_i^W(s) := \lambda_i(s)$ (as was proposed in [41]).

[1] When $\lambda_i(s) = \lambda^*$, one needs to decide for the action to be taken in state s in an appropriately randomized fashion that ensures that the relaxed constraint is satisfied [40, 41].

How do we find $\lambda_i(\cdot)$? Recall that the decision maker is indifferent when $\lambda = \lambda_i(s)$, and that we are interested in the case where the cost λ is chosen to be the optimal cost λ^* that causes the decision maker to select k arms on average. Furthermore, we saw that the Lagrangian (18.8) can be solved by considering arms one by one (in accordance with the intuition described at the beginning of this section, where it is argued that not much is lost by decoupling arms provided the system is large enough). In fact, (18.9) is the Lagrangian corresponding to a *one-arm sub-problem* in which there is only a single arm which can be selected or not, and where selecting the arm yields the state-dependent reward as well as an associated cost λ. Now the optimal $\lambda_i(s)$ is the one that makes us indifferent between selecting or not selecting the arm when it is in state s. In summary, we define the Whittle index as follows (cf. [41]).

Definition 18.2. The Whittle index is the largest cost λ in (18.9) such that it is still optimal to select the arm in the one-arm sub-problem.

Intuitively, the Whittle index can perhaps best be thought of as an opportunity cost, to be paid for loosing the opportunity to select one of the other arms in the constrained system with multiple arms. Naturally, we then prioritize arms with higher opportunity cost.

Computing the Whittle Index. As stated in Definition 18.2, the Whittle index is derived from the optimal policy for the one-arm sub-problem. Thus, the computational complexity of the Whittle index increases only linearly with the number of arms: we need the optimal policy for at most d non-identical single-arm sub-problems. In contrast, the complexity of computing the optimal policy for the full system increases exponentially (the latter problem is in fact PSPACE hard [33]).

It is well-known [14, 34] that in great generality the optimal average reward G is constant (independent of the initial state), and satisfies Bellman's optimality equation. For the one-arm sub-problem associated with arm i this optimality equation reads as,

$$G + H(s) = \max\left\{ r^i(s) - \lambda + \mathbb{E}\Big[H\big(\mathcal{A}_i(s,U)\big)\Big],\ \mathbb{E}\Big[H\big(\mathcal{P}_i(s,U)\big)\Big] \right\}, \qquad (18.10)$$

with $s \in S_i$ and U a uniform $(0,1)$ random variable. Here, $r^i(s) - \lambda$ is the immediate reward obtained from deciding to use the arm, corrected by the opportunity cost λ. The bias function H accounts for the transient effect caused by starting at initial state s rather than at equilibrium.

The optimal policy for this one-arm sub-problem is then to choose the action that maximizes the right-hand side of (18.10). It can be found from dynamic programming algorithms such as (relative) value or policy iteration. Then the Whittle index for state s can be effectively computed by solving (18.10) for an increasing sequence of λ and finding the maximal λ (namely $\lambda_i(s)$) for which selecting the arm is still optimal.

Note that for the RORMAB, the one armed subsidy problem (18.10) becomes:

$$G+H(s)=\max\left\{\mathbb{E}_s\left[R^i(X)\right]-\lambda+\mathbb{E}_s\left[H\left(\mathcal{O}(X)\right)\right],H\left(\mathcal{T}(s)\right)\right\}. \tag{18.11}$$

As before, an observation is obtained which is the realization of a random variable X with probability distribution determined by s (as indicated by the subscript). If the arm is not used, then no reward is obtained and the belief is propagated using the operator $\mathcal{T}$. We solve this problem numerically for GE and AR arms in the next section.

18.4 Numerical Illustration and Evaluation

We now return to RORMAB and compare the performance of the Whittle index policy to that of the myopic policy and, for small d, to the optimal policy. To evaluate the Whittle indices, we usually need the optimal policy associated with Whittle's one-armed problem with subsidy. We obtain the latter from relative value iteration (on a discretized state space) using the optimality equation (18.11). This can be written more explicitly using the reward functions from (18.5) as

$$G+H(\omega)=\max\left\{R^i(\omega)-\lambda+\omega H\left(p_i^{11}\right)+\overline{\omega}H(p_i^{01}),H\left(\omega p_i^{11}+\overline{\omega}p_i^{01}\right)\right\} \tag{18.12}$$

when the channel is GE (so that $s=\omega$, and $r^i(s)=R^i(\omega)$ since $R^i(\cdot)$ as defined in (18.5) is affine), and

$$G+H(\mu,\nu)=\max\left\{R^j(\mu)-\lambda+\int_{-\infty}^{\infty}H(\varphi y,\sigma^2)\,\phi_{\mu,\nu}(y)\,dy,H\left(\varphi\mu,\varphi^2\nu+\sigma^2\right)\right\} \tag{18.13}$$

when the channel is AR (in which case $s=(\mu,\nu)$, and $r^i(s)=R^i(\mu)$ since $R^i(\cdot)$ is linear). Here $\phi_{\mu,\nu}$ denotes the normal density with mean μ and variance ν. Note that in the case of GE channels, the Whittle indices are in fact available in closed form, [22]. Still, for the purposes of exposition, we carry out relative value iteration in this section numerically.

Figure 18.2 shows the optimal switching curve for a small mixed system with one AR and one GE channel. To the left of the curve, where ω is large in comparison to μ, the optimal policy is to select the GE channel. To the right of the curve selecting the AR channel is optimal. The curve shifts with the age of the AR channel: the more time has passed since the AR channel has last been observed, the more inclined the transmitter should be to select that channel in order to update the available information regarding its state. In other words, it is indeed optimal to give some priority to exploration if AR channels are present in the system. Note, however, that

for "older" channels this effect is less pronounced because in that case the resulting change in the conditional variance v is smaller (recall the belief propagation of v defined by (18.3)).

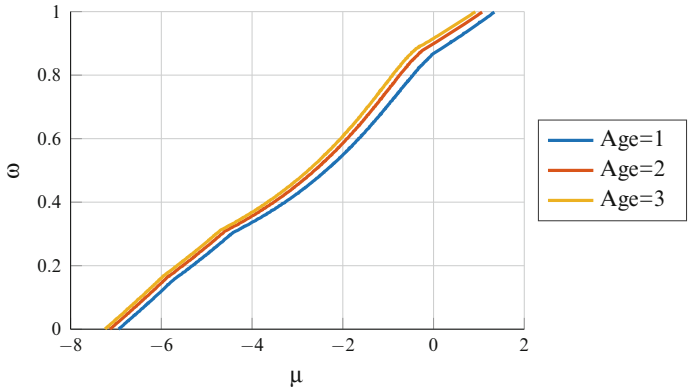


Fig. 18.2: Optimal switching curves for a system with $d = 2$ channels: an AR channel with $\varphi = 0.8$ and $\sigma = 2$, and a GE channel with $\rho = 0.5$ and $\gamma = 0.8$. This figure shows the switching curves on the ω, μ plane, one curve per age $\eta \in \{1,2,3\}$

Figure 18.3 shows a comparison of the rewards that are obtained per channel on average under different policies. Here, $k = d/2$ channels are selected at a time in a system with d channels, where half of the channels are GE and the other half is AR. All of the AR channels are with $\varphi = 0.8$ and $\sigma = 2$ (as in Fig. 18.6). The GE channels on the other hand are heterogeneous, with $\gamma = 0.8$ and $\rho_i \in [0.2, 0.8]$ evenly spaced such that $0.8 = \rho_1 > \cdots > \rho_{d/2} = 0.2$. Depicted are the average rewards per arm obtained under the Whittle and the myopic index policy, and, as an upper bound, we also computed the average rewards that could be obtained in a fully observable system under the myopic policy. Due to the high computational complexity, the optimal policy is only evaluated for $d = 2$.

All policies seem to approach a certain steady performance in terms of average reward per arm rather quickly as the number of channels grows large while the ratio k/d remains fixed. The achieved average reward level demonstrates the significant improvement in throughput that can be achieved by utilising the channel memory: the average reward is increased by more than 30% compared to the zero average reward that is obtained when channel memory is not used.

Figure 18.3 also confirms that some degree of exploration is favourable: In this example the Whittle index policy improves the average reward per arm by about 5% with respect to the myopic policy. This is in contrast to scenarios where all channels are GE and stochastically identical. In the latter case it can be shown that the Whittle and the myopic index policy are equivalent. We give further details in (i) below.

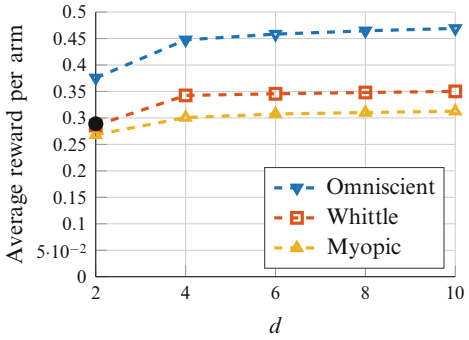


Fig. 18.3: Comparison of Whittle and myopic index policies for increasing number of channels d when half of the channels are GE and the other half is AR. For $d = 2$, the average reward obtained under the optimal policy is indicated by a black dot. We compare to the average reward that could be obtained if both arms were observed at each time point (that is in the fully observable or "omniscient" setting)

From a practical perspective an increase of 5% may appear small, however, it can be crucial in systems that are nearing fundamental limits. For example, for wireless devices with limited battery life such an increase may effectively correspond to a decrease of a few percent in power consumption, which may be significant in increasing the operational time of the device.

Next, we investigate the Whittle indices $\iota^W(\omega)$ obtained for GE channels with various parameter combinations (Fig. 18.4). We observe the following properties of $\iota^W(\omega)$:

1. The index function $\iota^W(\omega)$ increases monotonically; the larger the conditional probability that the channel is in a good state, the more priority should be given to that channel. This implies that the Whittle index is equivalent to the myopic policy in systems with *identical* channels, as we mentioned above.
2. $\iota^W(\omega)$ is affine within the ranges $\left[0, \min\{p^{01}, p^{11}\}\right]$ and $\left[\max\{p^{01}, p^{11}\}, 1\right]$. Further, it changes slope at γ.

These properties have been proven in [22] for GE channels with reward function $r(\omega)$ given by the identity function.

We further note that the Whittle indices are overall smaller if γ is larger because in this case the rewards are smaller (as R^j defined by (18.5) is decreasing in γ).

In Fig. 18.5 we show the difference between the Whittle and the myopic index function. It can be seen that the index functions are basically identical on $\left[0, \min\{p^{01}, p^{11}\}\right]$ and $\left[\max\{p^{01}, p^{11}\}, 1\right]$: In these regions exploration is not essential as it is rather certain that the state will evolve towards γ. Accordingly, we see that ι^W and ι^M do differ around γ, and on a larger interval to the left of γ.

Figure 18.6 depicts the difference between Whittle and myopic indices as obtained for an AR channel. The obtained indices increase with μ because the expected immediate reward is larger. Note that for increasing age the Whittle indices increase

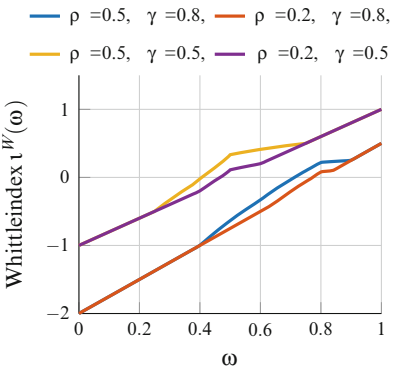


Fig. 18.4: Whittle indices for GE channels parametrized by α and γ

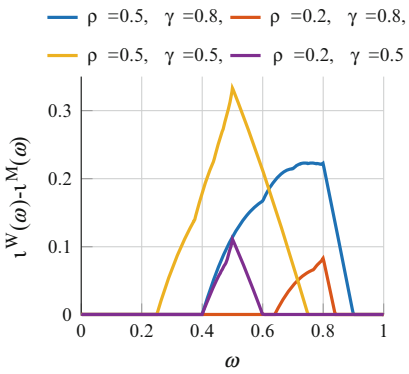


Fig. 18.5: Difference between Whittle and myopic index function

relative to the myopic indices. Again this suggests that exploration pays off. Furthermore, for high ages the difference between the Whittle and the myopic indices is largest around zero, which corresponds to the unconditional mean reward of the channel. Similarly to the GE case, this may be explained by noting that exploration is more important if μ is close to the unconditional mean as it is less clear in which direction the belief state will evolve. If μ is far away from the unconditional mean on the other hand, then it is likely that the updated conditional mean will be closer to the unconditional mean. However, when the age is close to zero, then due to the positive correlation of the channel it is also important that μ was large just an instance ago. Thus, while for small ages the Whittle indices are generally close to the myopic indices, the largest difference can be seen for positive μ (however not too far away from the unconditional mean of the channel).

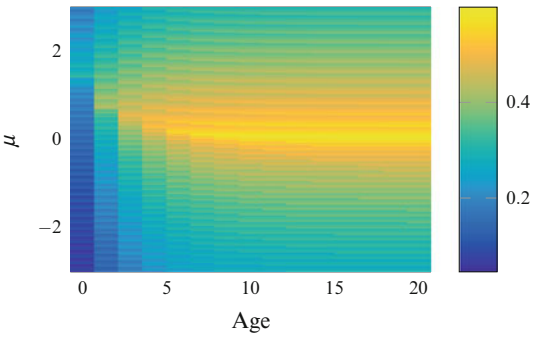


Fig. 18.6: Contour plot of $\iota^W(\mu, v) - \iota^M(\mu)$ (the difference of Whittle and myopic indices), for an AR channel with $\varphi = 0.8$, $\sigma = 2$

18.5 Literature Survey

There is a vast body of literature on MDP as well as topics related to (restless) multi-armed bandits. Here, we focus on the RORMAB formulation of the basic channel selection problem as formulated in this chapter, with GE or AR channels. Other (approximate) solutions to this MDP problem have been put forward [15], but are not considered here.

GE Channels. The Gilbert-Elliot model was proposed in [9] for the purpose of modelling burst-noise telephone circuits. It was the first non-trivial channel model with memory. Since the 1990s, the model and its generalizations have been used for modelling flat-fading channels in wireless communication networks. Its application in the context of Opportunistic Spectrum Access (OSA) is motivated by the bursty traffic of primary users [17, 45]. For an account on the history of the GE model we refer to [35].

Due to its simplicity, the GE model is mathematically tractable and has been analysed extensively in the context of channel selection in wireless networks. We survey a number of papers that model the problem as a RORMAB with GE channels. Unless otherwise stated, channels are assumed to be independent and stochastically identical.

One of the first papers in this context appears to be [19]. The paper is motivated by the problem of allocating bandwidth of a shared wireless channel between a base station and multiple homogeneous mobile users. Thus, from an engineering perspective, the set-up slightly differs from the problem considered in this chapter; the model and the mathematical analysis, however, apply directly to the channel selection scenario (where simply "users" are replaced by "channels").

In [19], the noisiness of the link for the users is modelled using the GE model. At any point in time a user may either be connected to the base station or not. The current state of a user is only observed when a packet is transmitted to that user. Rewards are given by the number of successful transmissions. The analysis is with respect to the *discounted* reward criterion over an infinite time horizon. The authors show that the myopic policy is generally optimal for the case of $d = 2$ users. For the case $d > 2$ and positively correlated channels it is proven that the myopic policy is optimal if the discount factor is small enough (Condition (A) in [19]). Furthermore, in the positively correlated scenario the myopic policy is seen to be equivalent to a "persistent round robin" policy where the link is dedicated to each user in a cyclic fashion according to their initial probability of being in a good state, and packets are transmitted to the same user until a packet fails to be transmitted correctly.

Following this work, the GE channel model has been analysed extensively in a surge of research on OSA, which goes back to [18]. The aim of this branch of research is to find secondary user policies that efficiently exploit transmission opportunities created by the bursty usage patterns of licensed primary users in wireless networks.

One of the first to formulate the RORMAB with GE channels in the context of OSA were Zhao et al. in 2005 [43]. The authors compare the transmission rate achieved by the myopic policy to the optimal policy using numerical examples.

This work was the starting point of a sequence of papers analysing the performance of the myopic policy. In [42], optimality is proven for the case of choosing one out of two channels, with respect to expected total discounted rewards over finite as well as infinite time horizon.

The scenarios in which the myopic policy is optimal are then generalized in a sequence of papers. Javidi et al. [16] consider the case of selecting 1 out of d channels and prove optimality of the myopic policy under the discounted reward criterion for positively correlated channels provided the discounted factor satisfies a certain inequality with respect to the transition probabilities. Under the additional ergodicity criterion

$$|p^{11} - p^{00}| < 1, \tag{18.14}$$

the myopic policy is further shown to be optimal under the average reward criterion (cf. (18.1)). The work of [16] is extended in [21] to the case of selecting $d-1$ out of d channels.

In [44] for the case of choosing 1 out of d channels the result of [19] is confirmed that the myopic policy is a persistent round robin scheme if channels are positively correlated. It is further shown that if correlation is negative, then the myopic policy is a round robin scheme, where the circular order is reversed in every time slot (and as for the positively correlated case, the user switches to the next channel as soon as the currently used channel signals has transitioned to the bad state). For the case $d = 2$, the myopic policy is shown to be optimal in general, as had already been established in [19]. Furthermore, it is shown that the performance of the myopic policy is determined by the stationary distributions of a higher-order countable-state Markov chain. The stationary distribution is known in closed form for the case $d = 2$. For the case $d > 2$, lower and upper bounds are established.

For negatively correlated channels and the case of selecting 1 out of d channels, the finite and infinite horizon discount-reward optimality of the myopic policy is proven in [3], provided that either $d \in \{2,3\}$ or the discount factor is less than half. These results also hold under average rewards under the additional ergodicity condition (18.14). For the finite-horizon discounted reward criterion, the results of [3] are generalized in [2] to the case of selecting k channels.

In 2014, Liu et al. [24] provide a unifying framework of the optimality conditions for the myopic policy that resulted from the OSA-motivated research of the channel selection problem with GE channels. The problem formulation in [24] is more general as it allows one to sense k out of d (identically distributed) channels but select only $l \leq k$ of those channels for transmission based on the outcome of the sensing. The authors provide a set of unifying sufficient conditions under which the myopic policy is optimal. It is shown that the optimal policy is not generally myopic if $l < k$. (This is intuitive because the user is allowed to explore channels without having to use them.)

The Whittle index policy has also been studied both for the bandwidth allocation problem that was put forward in [19], and also in the context of OSA. As opposed to [19], a paper by Niño-Mora [28] handles the problem of bandwidth allocation when users are *heterogeneous*. The author proves that the problem is indexable and provides closed-form expressions for the index function.

For the basic RORMAB with GE channels, Liu and Zhao [22] prove that the Whittle index and the myopic policy are equivalent for positively correlated identical channels, thus, yielding the optimality of the Whittle index in this case. In [32], the indexability and closed-form expression for the Whittle index in the case of discounted rewards are derived for a more general model where the achievable transmission rate (the reward) for a channel in the bad state is, in general, non-zero and any rate above this achievable rate leads to outage.

Apart from the index policies proposed in this line of research, algorithms for approximating an optimal policy have also been investigated. See, for example, [11, 13], where algorithms for the more general model with correlated channels are proposed and investigated regarding their performance.

In the context of GE channels a number of generalizations of the basic model considered in this chapter have been considered. For example, a paper by Niño-Mora [29] allows for non-identical channels with sensing errors/measurement noise. Imperfect sensing was also considered in [23, 39]. In [32] the authors consider a problem where in both states, good and bad, transmission may fail with a certain non-zero probability, and it is only observed whether transmission was successful or not. Another recent paper with imperfect sensing is [26]. In this paper the focus is on stability issues of queues associated with channel (server) selection in the context of imperfect sensing.

The paper [25] deals with random delay of feedback arrivals. Correlated channels were considered in [11–13]. Action-dependency of channel model parameters is taken into account in [37]. A very substantial paper is [38], which considers an RMAB in continuous time, and allows for non-identical channels, a time-dependent number of channels, and multiple actions. In this paper, a more general class of index policies is considered, which includes the Whittle index if the bandit problem is indexable. Asymptotic optimality for this class is proven for systems with many channels.

AR Channels. The AR channel model has only recently come to attention in the context of channel selection, and consequently the mathematical analysis is still at its starting point. The first to propose the application of this model for channel selection were Avrachenkov et al. [4] in 2012. This is motivated by empirical studies [1], showing that the AR model captures the signal-to-noise ratio (SNR) behaviour of the channels reasonably well.

In [4], the authors compare the performance of the myopic and an ad-hoc randomized policy to the optimal policy by means of numerical examples. It is concluded that the myopic policy is "nearly optimal" when all channels are similarly correlated, with respect to the long-run average reward criterion. In contrast, the

randomized policy appears to perform better when there is a significant difference in the magnitude of the correlation of the channels.

Subsequently, the authors show how to model the problem when two transmitters are present that can possibly interfere with each other. In this case the SNR is replaced by the signal-plus-interference-to-noise-ratio (SINR) to model the states of the channels. The scenario is formalized as a competitive MDP (also called a stochastic game)—an MDP in which the instantaneous rewards for each player and the transition probabilities among the states are controlled by the joint actions of the players in each state. Then, similar to the single user case, a randomized and a myopic policy are suggested (now based on the SINR).

A second paper that deals with channel selection with AR channels is [20]. This paper investigates structural properties of the Whittle index with respect to expected total discounted rewards. The monotonicity and convexity of the value function associated with the one-channel (arm) sub-problem is proven. Furthermore, numerical evidence for the indexability of the one-channel problem is provided, and the Whittle index policy is shown to outperform the myopic policy in numerical examples.

Then, a parametric index is proposed that is as simple as the myopic index but allows to give some priority to exploration, and therefore yields a better performance than the latter. For this parametric index, recursive equations are put forward that identify the asymptotic behaviour of the network in the setting with many channels. In addition, a simple heuristic algorithm is proposed to evaluate the performance of index policies; the latter is used to optimize the parametric index.

We also note that a related body of literature deals with the problem of optimal sensing of Kalman filters. A key paper in this line of research is [31]. A related paper is [30] as well as the recent [6] which appears to provide an indexability proof using a new novel method. It is possible that ideas put forward in these papers dealing with the Whittle index and simple Gaussian processes may be fruitful for the RORMAB problem with AR channels. This avenue of research remains to be explored.

Acknowledgements YN is supported by Australian Research Council (ARC) grants DP130100156 and DE130100291. JK is supported by DP130100156. The authors are indebted to Aapeli Vuorinen for his contribution to the numerical computations. We also thank the anonymous referee, Michel Mandjes and Thomas Taimre for their comments.

References

1. R. Aguero, M. Garcia, L. Munoz, BEAR: A bursty error auto-regressive model for indoor wireless environments, in *18th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)* (IEEE, New York, 2007), pp. 1–5
2. S.H.A. Ahmad, M. Liu, Multi-channel opportunistic access: a case of restless bandits with multiple plays, in *47th Annual Allerton Conference on Communication, Control, and Computing* (IEEE, New York, 2009), pp. 1361–1368
3. T.W. Archibald, D. Black, K.D. Glazebrook, Indexability and index heuristics for a simple class of inventory routing problems. Oper. Res. **57**(2), 314–326 (2009)

4. K. Avrachenkov, L. Cottatellucci, L. Maggi, Slow fading channel selection: a restless multi-armed bandit formulation, in *International Symposium on Wireless Communication Systems (ISWCS)* (IEEE, New York, 2012), pp. 1083–1087
5. D.P. Bertsekas, *Dynamic Programming and Optimal Control*, vol. 1 (Athena Scientific Belmont, 1995)
6. C.R. Dance, T. Silander, When are Kalman-filter restless bandits indexable? arXiv preprint arXiv:1509.04541, 2015
7. D. Duchamp, N. Reynolds, Measured performance of a wireless LAN, in *17th Conference on Local Computer Networks* (IEEE Press, New York, 1992), pp. 494–499
8. J. Gittins, K. Glazebrook, R. Weber, *Multi-Armed Bandit Allocation Indices*, 2 edn. (Wiley Online Library, Hoboken, 2011)
9. E.N. Gilbert, Capacity of a burst-noise channel. Bell Syst. Tech. J. **39**(5), 1253–1265 (1960)
10. J.C. Gittins, Bandit Processes and Dynamic Allocations. J. R. Stat. Soc. Ser. B Methodol. **41**(2), 148–177 (1979)
11. S. Guha, K. Munagala, Approximation algorithms for partial-information based stochastic control with Markovian rewards, in *48th Annual Symposium on Foundations of Computer Science (FOCS'07)* (IEEE, New York, 2007), pp. 483–493
12. S. Guha, K. Munagala, P. Shi, Approximation algorithms for restless bandit problems, in *ACM-SIAM Symposium on Discrete Algorithms (SODA)* (2009)
13. S. Guha, K. Munagala, P. Shi, Approximation algorithms for restless bandit problems. J. ACM **58**(1), 3 (2010)
14. O. Hernández-Lerma, J.B. Lasserre, *Discrete-Time Markov Control Processes: Basic Optimality Criteria*, vol. 30 (Springer, New York, 2012)
15. A. Itai, Z. Rosberg, A golden ratio control policy for a multiple-access channel. IEEE Trans. Autom. Control **29**(8), 712–718 (1984)
16. T. Javidi, B. Krishnamachari, Q. Zhao, M. Liu, Optimality of myopic sensing in multi-channel opportunistic access, in *International Conference on Communications (ICC'08)* (IEEE, New York, 2008), pp. 2107–2112
17. L.A. Johnston, V. Krishnamurthy, Opportunistic file transfer over a fading channel: a POMDP search theory formulation with optimal threshold policies. IEEE Trans. Wirel. Commun. **5**(2), 394–405 (2006)
18. R. Knopp, P.A. Humblet, Information capacity and power control in single-cell multiuser communications, in *International Conference on Communications (ICC'95)*, vol. 1 (IEEE, New York, 1995), pp. 331–335
19. G. Koole, Z. Liu, R. Righter, Optimal transmission policies for noisy channels. Oper. Res. **49**(6), 892–899 (2001)
20. J. Kuhn, M. Mandjes, Y. Nazarathy, Exploration vs exploitation with partially observable Gaussian autoregressive arms, in *8th International Conference on Performance Evaluation Methodologies and Tools (Valuetools)* (2014)
21. K. Liu, Q. Zhao, Channel probing for opportunistic access with multi-channel sensing, in *Asilomar Conference on Signals, Systems and Computers* (IEEE, New York, 2008)
22. K. Liu, Q. Zhao, Indexability of restless bandit problems and optimality of Whittle index for dynamic multichannel access. IEEE Trans. Inf. Theory **56**(11), 5547–5567 (2010)
23. K. Liu, Q. Zhao, B. Krishnamachari, Dynamic multichannel access with imperfect channel state detection. IEEE Trans. Signal Process. **58**(5), 2795–2808 (2010)
24. Y. Liu, M. Liu, S.H.A. Ahmad, Sufficient conditions on the optimality of myopic sensing in opportunistic channel access: a unifying framework. IEEE Trans. Inf. Theory **60**(8), 4922–4940 (2014)
25. S. Murugesan, P. Schniter, N.B. Shroff, Multiuser scheduling in a Markov-modeled downlink using randomly delayed arq feedback. IEEE Trans. Inf. Theory **58**(2), 1025–1042 (2012)
26. Y. Nazarathy, T. Taimre, A. Asanjarani, J. Kuhn, B. Patch, A. Vuorinen, The challenge of stabilizing control for queueing systems with unobservable server states, in *5th Australian Control Conference (AUCC)*, Nov 2015, pp. 342–347

27. J. Niño-Mora, Dynamic priority allocation via restless bandit marginal productivity indices. Top **15**(2), 161–198 (2007)
28. J. Niño-Mora, An index policy for dynamic fading-channel allocation to heterogeneous mobile users with partial observations, in *Next Generation Internet Networks (NGI)* (IEEE, New York, 2008), pp. 231–238 .
29. J. Niño-Mora, A restless bandit marginal productivity index for opportunistic spectrum access with sensing errors, in *Network Control and Optimization*, ed. by R. Núñez-Queija, J. Resing. Lecture Notes in Computer Science, vol. 5894 (Springer, Berlin, 2009), pp. 60–74
30. J. Niño-Mora, S.S. Villar, Multitarget tracking via restless bandit marginal productivity indices and Kalman filter in discrete time, in *Proceedings of the 48th IEEE Conference on Decision and Control, 2009 Held Jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009* (IEEE, New York, 2009), pp. 2905–2910
31. J.L. Ny, E. Feron, M. Dahleh et al., Scheduling continuous-time Kalman filters. IEEE Trans. Autom. Control **56**(6), 1381–1394 (2011)
32. W. Ouyang, S. Murugesan, A. Eryilmaz, N. B. Shroff, Exploiting channel memory for joint estimation and scheduling in downlink networks, in *30st Annual International Conference on Computer Communications (INFOCOM)* (IEEE, New York, 2011), pp. 3056–3064
33. C.H. Papadimitriou, J.N. Tsitsiklis, The complexity of optimal queuing network control. Math. Oper. Res. **24**(2), 293–305 (1999)
34. M.L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, vol. 414 (Wiley, New York, 2009)
35. P. Sadeghi, R.A. Kennedy, P.B. Rapajic, R. Shams, Finite-state Markov modeling of fading channels – a survey of principles and applications. IEEE Signal Process. Mag. **25**(5), 57–80 (2008)
36. R.D. Smallwood, E.J. Sondik, The optimal control of partially observable Markov processes over a finite horizon. Oper. Res. **21**(5), 1071–1088 (1973)
37. J.A. Taylor, J.L. Mathieu, Index policies for demand response. IEEE Trans. Power Syst. **29**(3), 1287–1295 (2014)
38. I.M. Verloop, Asymptotically optimal priority policies for indexable and non-indexable restless bandits. Ann. Probab. To appear. Retrieved 07/09/2015 from https://hal.archives-ouvertes.fr/hal-00743781.
39. K. Wang, L. Chen, Q. Liu, K. Al Agha, On optimality of myopic sensing policy with imperfect sensing in multi-channel opportunistic access. IEEE Trans. Commun. **61**(9), 3854–3862 (2013)
40. R. Weber, G. Weiss, On an index policy for restless bandits. J. Appl. Probab. **27**(3), 637–648 (1990)
41. P. Whittle, Restless bandits: activity allocation in a changing world. J. Appl. Probab. **25A**, 287–298 (1988). Special volume: A celebration of applied probability
42. Q. Zhao, B. Krishnamachari, Structure and optimality of myopic sensing for opportunistic spectrum access, in *International Conference on Communications (ICC'07)* (IEEE, New York, 2007)
43. Q. Zhao, L. Tong, A. Swami, Decentralized cognitive mac for dynamic spectrum access, in *First International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)* (IEEE, New York, 2005)
44. Q. Zhao, B. Krishnamachari, K. Liu, On myopic sensing for multi-channel opportunistic access: structure, optimality, and performance. IEEE Trans. Wirel. Commun. **7**(12), 5431–5440 (2008)
45. M. Zorzi, R.R. Rao, L.B. Milstein, Error statistics in data transmission over fading channels. IEEE Trans. Commun. **46**(11), 1468–1477 (1998)

Chapter 19
Flexible Staffing for Call Centers with Non-stationary Arrival Rates

Alex Roubos, Sandjai Bhulai, and Ger Koole

Abstract We consider a multi-period staffing problem of a single-skill call center. The call center is modeled as a multi-server queue in which the staffing levels can be changed only at specific moments in time. The objective is to set the staffing levels such that a service level constraint is met in the presence of time-varying arrival rates. We develop a Markov decision model to obtain time-dependent staffing levels for both the case where the arrival rate function is known as well as unknown. The characteristics of the optimal policies associated to the two cases are illustrated through a numerical study based on real-life data. We show that the optimal policies provide a good balance between staffing costs and the penalty probability for not meeting the service level.

Key words: Call centers, Markov decision processes, Staffing, Time-varying arrival rates

19.1 Introduction

Call centers have become the central focus of many companies, as these centers stay in direct contact with the firm's customers and form an integral part of their customer relationship management. Running a successful call center operation means managing by the numbers. One of the most important numbers in call centers is the

A. Roubos (✉)
CCmath, H.J.E. Wenckebachweg 48, 1096 AN Amsterdam, The Netherlands
e-mail: alex@ccmath.com

S. Bhulai • G. Koole
Faculty of Sciences, Vrije Universiteit Amsterdam, De Boelelaan 1081a,
1081 HV Amsterdam, The Netherlands
e-mail: s.bhulai@vu.nl; ger.koole@vu.nl

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_19

number of agents serving incoming calls at each moment of time. Since more than two-thirds of the operating costs can be attributed to personnel, getting the right number of agents in place is critical in terms of both the offered service and the operating costs. This agent staffing problem is a complex problem in which many issues have to be taken into account, e.g., demand forecasting, variability in the call arrival patterns, quality of service, and flexibility of the workforce. We refer the reader to the comprehensive surveys in [1, 12].

In this chapter, we consider the staffing problem in a single-skill call center for a given working day. The inherent randomness in the call center, due to variability in the duration of the calls and fluctuations in the call arrival rates, makes the staff problem complex. The randomness is the root cause of deviations of the performance measures from the predicted values at the moment of planning, see, e.g., [4, 16, 17, 19, 24, 31]. Traditionally, most call center literature assume known and constant mean arrival rates, mainly for the purpose of tractability. However, in addition to the usual uncertainty that is intrinsic to stochastic modeling, real call center data shows that there is also uncertainty in the process parameters. Since most performance indicators are sensitive to fluctuations in the parameters [18], both types of uncertainty should be accounted for in any staffing algorithm.

A substantial body of the literature has focused on the probability distribution of the arrival rates from a statistical perspective, see, e.g., [2, 4, 8, 9, 22, 26, 27, 29, 30]. These papers mostly deal with modeling the time-varying arrival process such that the essential features of call center arrivals are captured, e.g., a variance larger than the mean for the number of arrivals, a time-varying arrival intensity, and nonzero correlation between arrival counts in different periods.

Staffing in the presence of time-varying arrival processes was analyzed first by using the pointwise stationary approximation (PSA), see, e.g., [13, 15, 16], in which it is assumed that the arrival rates are known, deterministic, and non-stationary. However, the PSA does not explicitly consider non-stationary behavior that may be induced by abrupt changes in the arrival rate, and it appears to perform less well in these cases. Further numerical methods have been studied by Yoo [32], Ingolfsson et al. [18] and Feldman et al. [11]. The first two are based on methods that solve the Chapman-Kolmogorov forward equations by using small, discrete intervals to approximate the continuously varying parameter. The latter is based on an iterative simulation-based staffing method to achieve time-stable performance.

The case of unknown non-stationary arrival rates has been studied by Jongbloed and Koole [20], Steckley et al. [28], Harrison and Zeevi [17], Whitt [31], Robbins [24] and Liao et al. [21]. The first paper mainly focuses on the characterization of the uncertainty by providing bounds on the number of agents needed. The second paper studies the impact of different performance measures under this uncertainty. The next two papers focus on fluid approximations to determine the number of agents. The next paper uses simulation to derive the number of agents, whereas the last paper uses a robust programming formulation. Characteristic to these papers is that they study the staffing problem under uncertainty using a fixed staffing approach.

In contrast to these papers, in our problem there exists some flexibility to change the number of agents at fixed moments of the working day and thus creates more flexibility.

We model the arrival process as a non-stationary stochastic process with uncertain rates. Moreover, as is common in call centers, the call center operates under a service level that constrains the waiting time for incoming calls. The distinguishing feature of our model is that the staffing levels can only be changed at specific moments of the day, but still have to respect the service level constraint. We assume that there is a fixed number of employees with a permanent contract and a number of flexible agents that can be changed throughout the day on specific moments. The costs of using an agent differs between the fixed and flexible agents. The objective is to find the optimal staffing level that minimizes the total call center operating cost while meeting the service level constraint. We develop a Markov decision model that determines the optimal agent staffing policies in case the arrival rate function is both known and unknown. We conduct a numerical study in order to illustrate the main characteristics of the optimal solutions corresponding to these approaches. In the numerical illustration, we use real call center data and show how the optimal policy balances the staffing costs and the penalty probability for not reaching the service level. Furthermore, we show how the number of periods in which the staffing level can be changed affects the staffing costs.

The paper that is closest to our model is [22]. In this paper intraday updates of the call arrival rate are also allowed. The updates are based on the cumulative number of actual arrivals and the cumulative number of expected arrivals. The ratio between these numbers is used to adjust the forecast of the next intervals. Based on the update, the new staffing levels in each period are updated using the stationary independent period by period (SIPP) approach [14]. Moreover, the performance measure used in the paper is the expected service level. In our work, we do not only look at the expected service level, but at the whole distribution of the service level, and incur a penalty when the service level at the end of the planning period is not above a certain target. Hence, we need to address the effect of a change in the number of agents on the service level at the next decision epoch. Clearly, the SIPP approach is not sufficient to address this issue. Hence, we use a dynamic programming approach to assess this impact. It is this aspect of the problem that distinguishes our model from [22], but also from flexible staffing models in service facilities other than call centers, e.g., [5–7, 10, 23].

The remainder of the chapter is structured as follows. In Sect. 19.2, we describe the call center model under consideration and formulate the associated staffing problem. In Sect. 19.3 we formulate our staffing algorithm for the case of both known and unknown arrival rate functions. In Sect. 19.4, we conduct a numerical study to evaluate the alternative formulations. We illustrate the impact of the number of moments at which the staffing level can be changed, and thus the benefits of flexibility in the call center. The chapter ends in Sect. 19.5 with concluding remarks and highlights some future research.

19.2 Problem Formulation

Consider a call center to which customers arrive according to a non-homogeneous Poisson process with parameter λ_t for $t \geq 0$. We assume that the call center has s_t fixed permanent agents and f_t flexible agents at each time t and only N workplaces available so that $s_t + f_t \leq N$ for all $t \geq 0$. If upon arrival of a new customer at time t no agent out of the $s_t + f_t$ agents is available, then the customer joins an infinite buffer. In the other case, the customer is directly taken into service by an idle agent and has an exponentially distributed service duration with parameter μ_t. Queued customers are served in a first-come first-served order.

The objective of the call center manager is to meet a service level requirement by varying the number of flexible agents over the day. More precisely, divide the length of the day into m smaller intervals, each of length θ. We assume that the arrival rate function λ_t is constant over each interval and unknown. Hence, we also take s_t to be constant over each interval. Let SL_i represent the realized service level over interval $i = 1, \ldots, m$, given by the fraction of customers that has waited less than the acceptable waiting time τ upon starting service within that interval. The requirement of the call center is that SL, the service level over the whole day, is at least α, where SL is computed by the average of the SL_i's weighted by the arrival counts in each interval. The decision variable of the call center manager to achieve this requirement is the variable f_t that can only be changed at epochs determined at the start of certain intervals, namely at the start of interval $t \in \mathcal{T} = \{1, \kappa+1, 2\kappa+1, \ldots, m-\kappa+1\}$, where κ is a divisor of m. Hence, the variable f_t is fixed for a longer period than λ_t and s_t, and needs to take into account the variability inherent to these variables. This is especially challenging since the arrival rate function is not known.

The problem as described above is common in call centers. It is not realistic to assume that the number of flexible agents are changed continuously over time. The assumption that λ_t is constant over small time intervals is also not unrealistic, since this is usually the result of data estimation procedures that reliably approximate the true arrival rate function when the interval length is small. We assume that the permanent agents have a cost c_1 per unit of time for each agent, and that the flexible agents cost c_2 per unit of time for each agent, with $c_2 > c_1$. Note that for any given staffing policy, one cannot guarantee that $\mathrm{SL} \geq \alpha$ is always met at the end of the day, due to randomness. When the service level at the end of the day is not met, we impose that the call center manager incurs a penalty P. We model this service level constraint as a soft constraint. With these additional cost definitions the problem under study becomes

$$\min \sum_{i=1}^{m} (c_1 s_i \theta + c_2 f_i \theta) + P \mathbb{1}_{\{\mathrm{SL} < \alpha\}}$$

subject to

$$\begin{aligned} & f_t = f_{t+1} = \cdots = f_{t+\kappa-1}, && \forall t \in \mathcal{T}, \\ & s_t + f_t \leq N, && t = 1, \ldots, m, \\ & f_t \in \mathbb{N}_0, && t = 1, \ldots, m. \end{aligned}$$

19.3 Solution Approach

In order to solve the call center staffing problem, we cast the problem as a finite-horizon Markov decision problem on epochs $\mathcal{T}$. However, several simplifying approximations are required for purposes of implementation. We refer to Appendix for an exact formulation that solves the problem theoretically.

Let $\mathcal{X}$ denote the state space, where at epoch $t \in \mathcal{T}$ the state $x_t \in \mathcal{X}$ denotes the service level realized up to epoch t, i.e., $x_t = \sum_{i=1}^{t-1} \tilde{\lambda}_i \mathrm{SL}_i / \sum_{i=1}^{t-1} \tilde{\lambda}_i$ for $t \in \mathcal{T}$, where $\tilde{\lambda}_i$ is the value of λ_i derived from the observed arrival counts. Normally, the state space would be modeled by $[0,1]$, however, we discretize the state space to $\mathcal{X} = \{0, 1/\omega, 2/\omega, \ldots, 1\}$, where the parameter ω controls how well the continuous state space is approximated. The realized service level at each epoch is rounded down to the nearest value in the new state space.

Let the action space be denoted by $\mathcal{A}_t = \{0, \ldots, N - \bar{s}_t\}$, where $\bar{s}_t = \max\{s_t, s_{t+1}, \ldots, s_{t+\kappa-1}\}$. Action $a_t \in \mathcal{A}_t$ means that the call center manager schedules $a_t = a_{t+1} = \cdots = a_{t+\kappa-1}$ flexible agents at epoch t after observing x_t.

Note that the definition of the state space is such that the Markov property does not hold. Therefore, it is impossible to give exact transition probabilities. Given that the service level x_t is known, we simulate the system to obtain the service level $x_{t+\kappa}$, given that $s_t + a_t, \ldots, s_{t+\kappa-1} + a_{t+\kappa-1}$ agents are available. We assume that at the beginning of each interval t, the system with arrival rate λ_t, service rate μ_t and $s_t + a_t$ agents has reached stationarity, which is not an unrealistic assumption when changes in the dynamics are not too severe. Starting from a steady-state situation, we apply simulations for the duration of an interval to obtain the service level distribution. Then, by convoluting this distribution over the κ intervals, we derive the distribution for the next epoch. This approach has the advantage that we can simulate the transition probabilities up front for each combination of x_t and a_t. Hence, we can store a table with combinations of x_t and a_t that give $p_t(x_t, a_t, x_{t+\kappa})$, i.e., the probability of moving from state x_t to $x_{t+\kappa}$ when action a_t is chosen.

Finally, the direct costs are given by

$$c_t(x_t, a_t) = \sum_{i=t}^{t+\kappa-1} (c_1 s_i \theta + c_2 a_i \theta + P \mathbb{1}_{\{i=m\}} \mathbb{1}_{\{\mathrm{SL} < \alpha\}}).$$

The first and second terms are related to the staffing of permanent and flexible agents. The last term corresponds to the penalty P that is incurred if the service level at the end of the day is not met.

The tuple $(\mathcal{X}, \mathcal{A}, p, c)$ completely describes the Markov decision process for this problem.

Note that in the problem above, the values of s_t are given. However, in practice, the values for s_t would be obtained by having an estimate of the values of λ_t for the specific day. This would typically be done in light of long-term personnel planning by using the Erlang C formula. The decision variable a_t can then be seen as short-term planning that adjusts for deviations on this estimate. It is worthwhile to mention that the use of the Erlang C formula for deriving values for s_t is not optimal in general (see Sect. 19.4 for some examples), but provides a good starting point for the staffing problem at hand.

In the description of our algorithm, we mention that at epoch t we define the state x_t as the realized service level up to epoch t. This service level is easy to compute, since the arrival counts in intervals $i < t$ are known. At epoch t we also need the values λ_i for $i \geq t$ to determine the optimal actions. However, these values are unknown and need to be obtained via an estimation procedure. Note that this estimation procedure can be different than the procedure used to determine the values of s_t, since the realized values up to epoch t can be used as well and provide better information on the future values of λ_t. Examples of estimation procedures can be found in, e.g., [2, 4, 27].

19.4 Numerical Experiments

In this section we show the characteristics of the optimal policies by means of numerical experiments. The parameters of the experiments are based on real-life data, or otherwise chosen to represent parameters that can be found in practice. We start with an example that demonstrates the benefits of the flexibility in staffing. This example assumes a known and constant arrival rate.

Remark 19.1. In order to evaluate the optimal policies, we apply independent simulations. We mainly focus on two performance measures: the total staffing costs and the probability that a penalty is incurred if at the end of the day the service level is lower than the target. Because the penalty probability is extremely small, many simulations are necessary to obtain an accurate estimate, see, e.g., the topic of rare-event simulation in [3]. For instance, for a probability $p = 0.01$ and a 95% confidence interval with half-width equal to $0.1p$, the number of simulations should be at least $n = 40{,}000$. We perform $n = 1{,}000{,}000$ simulations, which can be calculated within a few seconds. We present the half-width of the 95% confidence intervals between parentheses, but omit confidence intervals for values that have a negligible half-width.

19.4.1 Constant Arrival Rate

Consider a call center with no flexibility, i.e., only a fixed number of agents are scheduled for the whole time horizon. The arrival rate is $\lambda = 3$ per minute and the service rate is $\mu = 0.2$ per minute. The acceptable waiting time is $\tau = 1/3$ min (20 s). Based on these parameters, the Erlang C formula tells us that we need $s = 19$ agents in order to meet the 80% service level target. Each agent costs $c_1 = 1$ unit per minute. Suppose the call center operates for a time horizon of $T = 720$ minutes (12 h). The call center starts empty, and waiting customers at the end of the time horizon are ignored.

With these parameters the costs for staffing are $C = 1 \times 19 \times 720 = 13{,}680$. However, despite the fact that the expected service level is above 80% as predicted by the Erlang C formula, simulations show that the realized service level is in many cases below 80%. With probability 0.34 the service level falls below the required target and hence a penalty is incurred. This phenomenon that the service level is not always reached in a finite-time horizon is discussed in [25]. One way to deal with this problem, without flexibility, is to try a higher staffing level. With $s = 20$ the costs for staffing are increased to $C = 14{,}400$. Furthermore, the probability that a penalty is incurred is reduced to only 0.03.

We now allow flexible agents. We have a base staffing level of $s = 19$ for the whole time horizon. Each 30 min there is the opportunity to add additional flexible agents (i.e., $m = 24$, $\theta = 30$ and $\kappa = 1$), against a cost of $c_2 = 1.2$ units per minute per agent. When choosing to do so, the extra agents are immediately available. We take $N = 30$, which is sufficiently large for this example. Furthermore, we incur a sufficiently high penalty of $P = 10^6$ for failing to meet the target service level at the end of the day. The state space is discretized according to $\omega = 400$. The transition probabilities $p_t(x_t, a_t, x_{t+\kappa})$ are determined from 10,000 sample path simulations for each action a_t. By application of our method we find the optimal policy. Evaluation by means of simulations shows that the costs for staffing are $C = 14{,}192$ and that the probability of a penalty is 0.0018 (8.4×10^{-5}). This is a considerable improvement in both staffing costs and penalty probability compared to staffing 20 agents with no flexibility.

The optimal policy is displayed in the left plot in Fig. 19.1. This figure should be interpreted as follows. For a decision moment at time t, and given a realized service level up to t, the number of flexible agents staffed is given by the figure. The optimal policy suggests that in some cases 30 agents should be scheduled (a base staffing level of 19 plus 11 flexible agents). With that many agents the probability to reach a service level of 1 in a 30-minute interval is already 0.996. The large white area below the curve denotes pairs of time epochs and service levels that always result in an expected service level lower than the target. Hence, no flexible agents are staffed.

The right plot in Fig. 19.1 shows boxplots of the service level at a decision epoch. The small circles are outliers, where an outlier is a data value more extreme than 1.5 times the interquartile range from the box. This figure shows that the average service

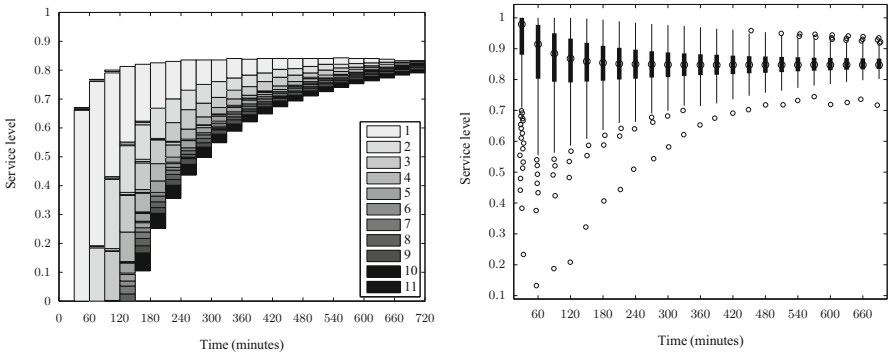


Fig. 19.1: *Left plot*: the optimal policy for the example with a constant arrival rate. *Right plot*: boxplots of the service level at a decision epoch

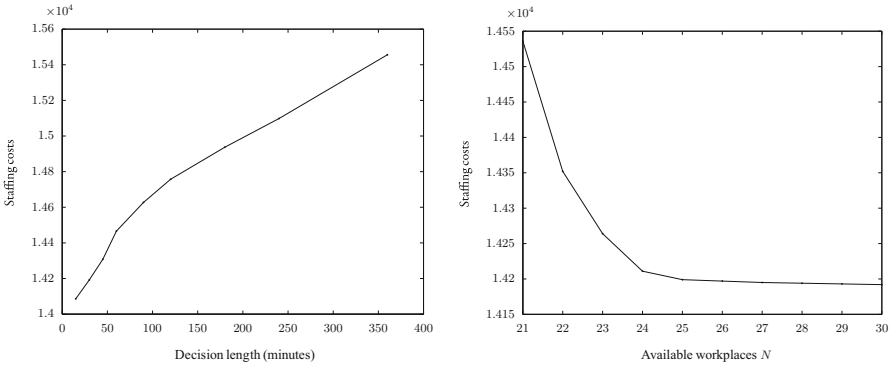


Fig. 19.2: Value of flexibility

level nicely converges to approximately 0.85. Moreover, the variability greatly diminishes over time, and outliers are becoming more sparse.

19.4.1.1 Value of Flexibility

It is clear that staffing only a few flexible agents at the right moments keeps both the staffing costs and the penalty probability low. In the previous example we could vary the number of staffed flexible agents each 30 min. Allowing flexibility on this time scale might not be possible for all call centers. Also, the optimal policy was not restricted by the number of available workplaces. Therefore, we are interested in the effect of different levels of flexibility on the performance measures.

Figure 19.2 shows how the staffing costs depend on the frequency with which decisions can be made and on the number of available workplaces. The staffing costs in the left plot increase up to a maximum of $C = 15{,}840$, which is attained at staffing

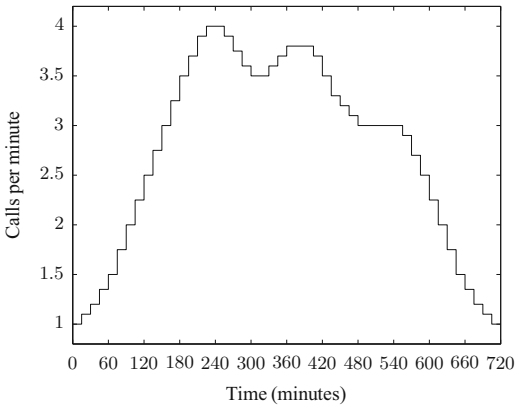


Fig. 19.3: The time-dependent arrival rate

22 agents for the whole day of 720 min. This plot shows that large improvements can be obtained if the call center can react on a small time scale. But there is also a significant gain if the call center can only adjust the staffing level once a day, after 360 min, since this reduces the staffing costs to $C = 15{,}456$. The right plot shows that most of the improvement comes from the first few flexible agents. The plot starts from a minimum of 21 workplaces, because with only 20 available workplaces there will always be a considerable penalty probability of 3%. With 21 workplaces, the staffing costs will be relatively high compared to a larger number of available workplaces, since the flexible agents are almost always used.

19.4.2 Time-Dependent Arrival Rate

We now consider a call center with a time-dependent arrival rate. We still assume that the arrival rate is known. In Fig. 19.3 the typical pattern of arrivals over the day is depicted. Here we model the arrival rate as a piecewise constant function, where each interval equals 15 min. All other parameters related to the model remain the same. Based on the stationary Erlang C formula, we find the base staffing level in each interval such that the target will be met. These staffing levels have the same shape as the arrival rate. Performance assessment concludes that with no flexibility the staffing costs are $C = 12{,}855$ and that the probability of failing to meet the target service level at the end of the day is 0.18. When staffing one agent more in each interval, the penalty probability is reduced to 0.01, but the staffing costs are then $C = 13{,}575$.

With the opportunity to add flexible agents we can improve this situation. We assume that decisions about flexible agents can only be made each consecutive 30 min, and that we have a limited number of available workplaces of $N = 30$. This implies, e.g., that we can only choose up to 5 flexible agents in the time periods $[210, 240)$

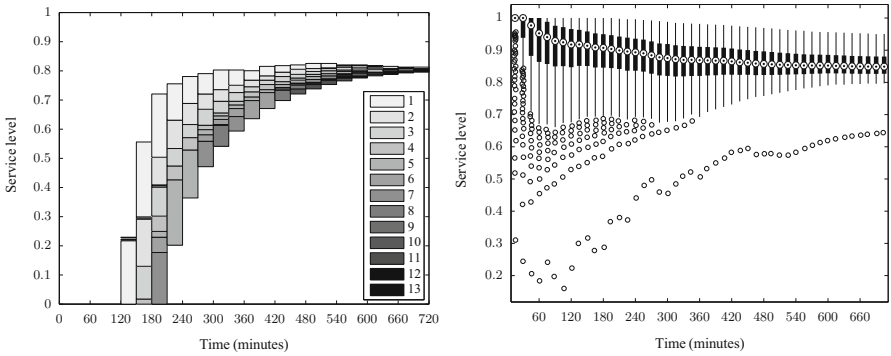


Fig. 19.4: *Left plot*: the optimal policy for the example with a time-dependent arrival rate. *Right plot*: boxplots of the service level at a staffing period

and $[240, 270)$, since there are already 25 permanent agents scheduled at $[225, 255)$. All other parameters related to our method and performance assessment remain the same. When we apply our method we find the optimal policy as shown in Fig. 19.4. The corresponding costs for staffing are $C = 13{,}101$ and the probability of a penalty is 0.0062 (1.5×10^{-4}). Again, this is a considerable improvement. It is astonishing to notice that this performance can be achieved by requiring on average 6.8 flexible agents at the right moments, which corresponds to only 3.4 agent hours.

The optimal policy reveals a very interesting characteristic. Until 120 min, no flexible agents are needed at all. This is, of course, due to the low arrival rates at the beginning of the day, which means that the realized service level up to 120 min is not that important. Consequently, this provides an excellent opportunity to better estimate the arrival rate in the remainder of the day, in case the arrival rate is unknown.

The right plot in Fig. 19.4 shows boxplots of the service level at the beginning of each 15 min staffing period. Most notably from this figure is that the whiskers extending from the bottom of the boxes are becoming shorter. There are hardly any realized service levels below the 80% service level target at the end of the day, which demonstrates that our method works well for this example.

19.4.2.1 Optimal Permanent Agents

Although the scope of this chapter is on flexible agents, we do make a short remark about the choice of the permanent agents. The flexibility of adding agents at the decision epochs can, and should, be taken into account when making a long-term planning of the permanent agents. Our method can also be used to do this. Consider for example the following heuristic approach. Start with the vector s such that $s_i = \min\{k \in \mathbb{N} \,|\, k > \lambda_i/\mu_i\}$ for $i = 1, \ldots, m$. That is, the number of agents in each interval is higher than the offered load, but as small as possible. Apply our

Table 19.1: The number of permanent agents

Base	8, 9, 9, 10, 11, 12, 14, 15, 17, 18, 19, 21, 22, 23, 24, 25, 25, 24, 23, 23, 22, 22, 23, 23, 24, 24, 24, 23, 22, 21, 20, 20, 19, 19, 19, 19, 19, 19, 18, 17, 15, 14, 12, 11, 10, 9, 9, 8
Optimal	7, 7, 8, 9, 10, 11, 13, 15, 16, 17, 19, 21, 23, 24, 25, 25, 25, 25, 24, 23, 22, 22, 23, 23, 23, 23, 24, 23, 22, 21, 20, 19, 18, 18, 18, 18, 19, 18, 16, 15, 13, 11, 9, 8, 7, 6, 6, 6

method to find the policy $\pi(s)$ and the corresponding costs $C^{\pi(s)}$. The next step consists of adding an additional permanent agent to exactly one interval, namely the interval that will result in the largest decline in costs. Let $s+e_i$ denote the vector with an additional agent at interval i, and let $j = \arg\min_{i=1,\ldots,m} C^{\pi(s+e_i)}$. Then, if $C^{\pi(s+e_j)} < C^{\pi(s)}$, update s to $s+e_j$. Continue this iteration until no improvement can be found anymore.

We use this heuristic approach on the previous example with the time-dependent arrival rate. All parameters remain the same, with the exception of P. We increased the penalty to $P = 10^{12}$ in order to keep the penalty probability low. We find the optimal policy similar to the one in Fig. 19.4. However, due to an overall decrease in the number of permanent agents, the staffing costs turn out to be much lower. They are $C = 12{,}935$, and the penalty probability is 0.0099 (1.9×10^{-4}). More flexible agents are used now, in order to reach the target service level. On average 18.5 flexible agents are needed for specific 30-minute intervals per day.

In Table 19.1 the number of permanent agents are given, for each interval of 15 min. This table compares the heuristic optimal staffing levels with the base staffing levels, where in each interval the target service level will be met according to the Erlang C formula. The optimal levels are for the most part lower, indicating that the availability of flexible agents is better utilized when necessary. Staffing is higher in five intervals with a high arrival rate. This ensures a higher expected service level in these intervals, and possibly compensating for other intervals of lesser importance. It is interesting to note that the last couple of intervals are really understaffed. This is due to the fact that the average service level can only be changed very limitedly near the end of the day.

19.4.3 Unknown Arrival Rate

In most practical situations the real arrival rate λ_t is not known. What is available is a best estimate $\hat{\lambda}_t$ that is estimated or forecast from historical data. It goes without saying that if this estimate is accurate ($\hat{\lambda}_t$ is close to λ_t) our method works well, in the sense that the service level requirement will always be achieved, because it

reduces to the case of a known arrival rate. What we are interested in is the performance in case the arrival rate estimate is inaccurate.

As more information becomes available over the course of the day, our algorithm updates the arrival rate estimate. In practice this can be done quite accurately, since a large database with historical arrival rates are available, and sophisticated updating procedures can be used (see, e.g., [27]). However, what we will show is that even with no knowledge of previous arrival rates, and therefore using a very basic updating method, our algorithm works just as well. The updating method we consider is the historical proportion method [27], which works as follows. At decision epoch $t \in \mathcal{T}$ calculate the ratio R between the realized and estimated arrival rate up to t, i.e., $R = \sum_{i=1}^{t-1} \tilde{\lambda}_i / \sum_{i=1}^{t-1} \hat{\lambda}_i$. Then, update the estimate for the remainder of the day: $\hat{\lambda}_i' = R\hat{\lambda}_i$, $i = t, \ldots, m$. This new estimate, together with the realized arrival rate, is then used to give an updated optimal policy.

As a result of this updating procedure, we need to evaluate the optimal policy multiple times per day. This computation takes roughly 1 min to carry out, namely we update 24 times a day for a calculation that runs for approximately a few seconds. Hence, an accurate evaluation by means of extensive simulations becomes hardly doable if $n = 1{,}000{,}000$ (see also Remark 19.1). Therefore, we have to settle for less accuracy in our simulations with $n = 1{,}000$. Also, the state space is now discretized according to $\omega = 200$. As in the examples before, we take $N = 30$, a penalty of $P = 10^6$ and allow flexible agents each consecutive 30 min.

In our experiments, we consider several cases with respect to the pattern of the arrival rate. In the first example the estimated arrival rate is the real arrival rate multiplied by a constant scalar, $\hat{\lambda}_t = \lambda_t \cdot \beta$. In the second and third examples we correctly estimate the arrival rate pattern, but we make a fixed under- or overestimation, $\hat{\lambda}_t = \lambda_t + \beta$, with $\beta = -0.5$ and $\beta = 0.5$. Finally, the fourth and fifth examples are examples with a wrongly estimated pattern, $\hat{\lambda}_t = \lambda_t \cdot \beta_t$, with $\beta_t = 1 - 0.005t$ and $\beta_t = 1 + 0.005t$. That is, the estimate becomes increasingly more wrong. In all examples, the true arrival rate is the one shown in Fig. 19.3.

For a fair comparison between the performance of the different examples we use the same number of permanent agents in each interval across the examples, which is the number determined by the Erlang C formula using λ_t, $\mu = 0.2$, $\tau = 1/3$ and $\alpha = 0.8$ in each interval (i.e., the base staffing levels). A reason against using the Erlang C formula with $\hat{\lambda}_t$ is that in the overestimated situations the server costs would be high and the penalty probabilities low, even without using flexible agents. Moreover, from the previous examples we have seen that the base staffing levels do require flexible agents in order to balance the server costs and the penalty probability.

The results of the experiments are shown in Table 19.2. The results for the first example are independent of β, because the β disappears in the updated estimate after the first epoch. As the day progresses, the estimate for the remainder of the day naturally becomes more accurate. Hence, this example can be seen in light of the previous example with a known and time-dependent arrival rate, though with more uncertainty. The results are also very similar. The underestimation of the arrival rate in the second example actually becomes an overestimation, because of the updating

Table 19.2: Results of experiments with an unknown and time-varying arrival rate

Example	Service level	Server costs	Penalty probability
1	0.853 (0.002)	13,100 (24)	0.024 (0.009)
2	0.874 (0.002)	13,903 (31)	0.018 (0.008)
3	0.855 (0.002)	13,140 (30)	0.007 (0.005)
4	0.851 (0.002)	13,067 (28)	0.031 (0.011)
5	0.865 (0.002)	13,245 (25)	0.004 (0.004)

method. Therefore, more flexible agents are used resulting in higher server costs, a higher service level and a decrease in penalty probability. The third example is exactly the opposite, in the sense that for most intervals the arrival rate will be underestimated. However, an overestimation will still happen in the last ten intervals. This example shows that our method works by adapting only when the service level is too low. That the penalty probability is low, is due to the overestimation in the intervals at the end. Examples four and five show results as could be expected for under- and overestimated arrival rates. That the penalty probability is not equal to zero is again due to the approximate transition probabilities.

19.5 Conclusion and Discussion

In this chapter we have shown that significant improvements can be obtained by introducing flexible agents. The improvements are expressed in the form of lower staffing costs or a lower probability of failing to meet the service level target at the end of the day, compared to the traditional approach that does not exploit this flexibility. Numerical experiments showed that our approach works remarkably well, even in the case of an unknown and time-varying arrival rate, with a forecast that is not necessarily accurate.

We model the call center as a Markov decision process in a non-traditional manner where our state variable denotes the service level as opposed to the number of customers in the system. The transition probabilities are, due to the complexities of calculating them exactly, obtained via simulations. This allows us to look further than (non-homogeneous) Poisson arrivals and exponential service times. As more information becomes available over the course of the day, we make use of a better estimated arrival rate to update the optimal policy. In the same way, we can also update the service time distribution. The case of agent absenteeism (e.g., a permanent agent is scheduled to work, but did not show up) is easily handled by decreasing the number of permanent agents s_t. The absent agent will be taken care of by a flexible agent, if that turns out to be necessary.

Our approach is highly relevant to call center practice. Uncertainty in call arrivals demands flexibility from a call center to guarantee good performance without incurring excessive staffing costs. In practice, many call centers indeed have this

flexibility. Flexibility in the workforce is achieved by, e.g., managers that help answering telephone calls during busy periods, or due to people that are flexible in their working hours, and can be requested to work on an ad-hoc basis with flexible contracts, such as students and agents that work from home. Additional flexibility can be obtained at the moment a shift of an agent ends and that agent can be requested to work overtime. This is practically relevant, since we observe that the demand for flexible agents increases at the end of the day, see Fig. 19.4. The algorithm in this chapter exploits this flexibility in call centers in an easily implementable fashion, and therefore has the potential to be integrated in workforce management software of call centers.

Appendix: Exact Solution

In this section, we formulate a discrete-time Markov decision problem for our original continuous-time problem. We only discretize time into small intervals, but make no other approximation. Hence, the formulation is nearly exact for small time intervals, and thus computes nearly optimal policies for the original problem. We denote the length of a time interval by $1/\eta$, thus every $1/\eta$ time units the system is observed.

In order to model the transitions of the system after each observation, we need a large state space that contains all information to calculate the next state. Hence, define the state space $\mathcal{X}$ to consist of tuples $(n, s_c, s_d, m, z, w_1, \ldots, w_n)$. In this tuple, $n \in \mathbb{N}_0$ denotes the number of customers in the system at the time of an observation. The realized waiting times of each of the n customers at the moment of observation is given by $w_1, \ldots, w_n \in \mathbb{R}_0^+$. We will adopt the convention that customers in service have a waiting time of 0. Further, let $s_c \in \mathbb{N}_0$ denote the number of servers currently in use, and $s_d \in \mathbb{N}_0$ the number of servers that is desired to have. The service level can be computed by the ratio of $z \in \mathbb{N}_0$, the number of customers served within τ time units, and $m \in \mathbb{N}_0$, the number of customers served. These variables are sufficient to model the state transitions in a Markovian way. Hence, the dynamic programming backward recursion formula becomes

$$
\begin{aligned}
\eta V_{k+1}(n, s_c, s_d, m, z, w_1, \ldots, w_n) &= c_1 s_k + c_2(s_c - s_k) \\
&+ \lambda \mathbb{1}_{\{s_c < n\}} H_k(n+1, s_c, s_d, m, z, w_1, \ldots, w_{s_c}, w_{s_c+1} + 1/\eta, \ldots, w_n + 1/\eta, 0) \\
&+ \lambda \mathbb{1}_{\{s_c = n\}} H_k(n+1, s_c, s_d, m, z, w_1, \ldots, w_n, 0) \\
&+ \lambda \mathbb{1}_{\{s_c > n\}} H_k(n+1, s_c, s_d, m+1, z+1, w_1, \ldots, w_n, 0) \\
&+ \mu s_c \mathbb{1}_{\{s_c = s_d\}} \mathbb{1}_{\{s_c < n\}} \big[H_k(n-1, s_c, s_d, m+1, z + \mathbb{1}_{\{w_{s_c+1} + 1/\eta < \tau\}}, \\
&\quad w_2, \ldots, w_{s_c}, 0, w_{s_c+2} + 1/\eta, \ldots, w_n + 1/\eta) \big] \\
&+ \mu s_c \mathbb{1}_{\{s_c > s_d\}} \mathbb{1}_{\{s_c < n\}} \big[H_k(n-1, s_c - 1, s_d, m, z, \\
&\quad w_2, \ldots, w_{s_c}, w_{s_c+1} + 1/\eta, \ldots, w_n + 1/\eta) \big]
\end{aligned}
$$

$$
\begin{aligned}
&+\mu n \mathbb{1}_{\{s_c=s_d\}} \mathbb{1}_{\{s_c \geq n\}} \left[H_k(n-1, s_c, s_d, m, z, w_2, \ldots, w_n)\right] \\
&+\mu n \mathbb{1}_{\{s_c>s_d\}} \mathbb{1}_{\{s_c \geq n\}} \left[H_k(n-1, s_c-1, s_d, m, z, w_2, \ldots, w_n)\right] \\
&+\left(\eta-\lambda-\min\{n, s_c\}\mu\right)\left[\mathbb{1}_{\{s_c \geq n\}} H_k(n, s_c, s_d, m, z, w_1, \ldots, w_n)\right. \\
&\left.+\mathbb{1}_{\{s_c<n\}} H_k(n, s_c, s_d, m, z, w_1, \ldots, w_{s_c}, w_{s_c+1}+1/\eta, \ldots, w_n+1/\eta)\right].
\end{aligned}
$$

The index k counts the number of intervals to go until the end of the complete period, the last interval. The first two terms describe the cost of using s_k permanent and $s_c - s_k$ flexible agents. If upon arrival, the number of servers currently in use is less than n, then there are $n - s_c$ customers in the queue. Hence, these customers add $1/\eta$ time units to their waiting time (term 3). If $s_c = n$, then everyone is in service, and the arriving customer has to wait (term 4). If $s_c > n$, then there are idle servers. Hence, an arriving customer is served immediately and satisfies the service level directly as well (term 5). The next two terms, terms 6 and 7, model the case where a customer leaves the system when there are customers waiting in the queue. The first case is where the number of servers currently in use is equal to the desired number. Hence, $1/\eta$ is added to the waiting times and s_c remains unchanged. When the customer is taken into service, then the service level is also adjusted. The second case is when s_c is higher than s_d, then additionally s_c is decreased by one and no customer is taken into service (thus, the service level is not updated either). Terms 8 and 9 model a similar situation, however, in this case there are a sufficient number of servers available so that no customer is waiting. Hence, the waiting times are not adjusted and neither is the service level. The final terms deal with the similar cases in which no event occurs within the interval. Hence, only the waiting times are updated when $s_c < n$.

In the dynamic programming backward recursion formula, we have adopted the notation H for the action operator. This action operator is equal to V for all intervals k that are not a decision epoch, i.e., $k \notin \mathcal{T}$. However, for all $k \in \mathcal{T}$, we have that

$$
\begin{aligned}
H_k(n, s_c, s_d, m, z, w_1, \ldots, w_n) = \min_{l \in \mathbb{N}_0} \big\{ &\{V_k(n, s_c, l, m, z, w_1, \ldots, w_n) \,|\, l < s_c\} \\
&\cup \{V_k(n, l, l, m, z, w_1, \ldots, w_n) \,|\, l \geq s_c\}\big\}.
\end{aligned}
$$

The first set in the minimization models the case in which the number of servers is decreased, hence s_d is adjusted. The second set models the case in which the number of servers is increased. Since this happens immediately, both s_c and s_d are set to the desired level.

Finally, we finish the model by describing what happens at the last interval. In this case, we can evaluate the realized service level and compare it to α. If the service level is not met, then a penalty of P is incurred, and otherwise no additional cost is incurred. This is given by the following equation.

$$
\eta V_0(n, s_c, s_d, m, z, w_1, \ldots, w_n) = c_1 s_k + c_2(s_c - s_k) + P \mathbb{1}_{\{z/m<\alpha\}}.
$$

References

1. O.Z. Akşin, M. Armony, V. Mehrotra, The modern call center: a multi-disciplinary perspective on operations management research. Prod. Oper. Manag. **16**(6), 665–688 (2007)
2. S. Aldor-Noiman, P.D. Feigin, A. Mandelbaum, Workload forecasting for a call center: methodology and a case study. Ann. Appl. Stat. **3**(4), 1403–1447 (2009)
3. S. Asmussen, P.W. Glynn, *Stochastic Simulation: Algorithms and Analysis* (Springer, New York, 2007)
4. A.N. Avramidis, A. Deslauriers, P. L'Ecuyer, Modeling daily arrivals to a telephone call center. Manag. Sci. **50**(7), 896–908 (2004)
5. J. Bard, H. Purnomo, Short-term nurse scheduling in response to daily fluctuations in supply and demand. Health Care Manag. Sci. **8**(4), 315–324 (2005)
6. R. Batta, O. Berman, Q. Wang, Balancing staffing and switching costs in a service center with flexible servers. Eur. J. Oper. Res. **177**(2), 924–938 (2007)
7. O. Berman, R.C. Larson, A queueing control model for retail services having back room operations and cross-trained workers. Comput. Oper. Res. **31**(2), 201–222 (2004)
8. L. Brown, A. Mandelbaum, A. Sakov, H. Shen, S. Zeltyn, and L. Zhao. Multi-factor Poisson and gamma-Poisson models for call center arrival times. Working Paper, 2004
9. L.D. Brown, N. Gans, A. Mandelbaum, A. Sakov, H. Shen, S. Zeltyn, L. Zhao, Statistical analysis of a telephone call center: a queueing-science perspective. J. Am. Stat. Assoc. **100**(469), 36–50 (2005)
10. F.F. Easton, J.C. Goodale, Schedule recovery: unplanned absences in service operations. Decis. Sci. **36**(3), 459–488 (2005)
11. Z. Feldman, A. Mandelbaum, W.A. Massey, W. Whitt, Staffing of time-varying queues to achieve time-stable performance. Manag. Sci. **54**(2), 324–338 (2008)
12. N. Gans, G.M. Koole, A. Mandelbaum, Telephone call centers: tutorial, review, and research prospects. Manuf. Serv. Oper. Manag. **5**(2), 79–141 (2003)
13. L.V. Green, P.J. Kolesar, The pointwise stationary approximation for queues with nonstationary arrivals. Manag. Sci. **37**(1), 84–97 (1991)
14. L.V. Green, P.J. Kolesar, J. Soares, Improving the SIPP approach for staffing service systems that have cyclic demands. Oper. Res. **49**(4), 549–564 (2001)
15. L.V. Green, P.J. Kolesar, J. Soares, An improved heuristic for staffing telephone call centers with limited operating hours. Prod. Oper. Manag. **12**(1), 46–61 (2003)
16. L.V. Green, P.J. Kolesar, W. Whitt, Coping with time-varying demand when setting staffing requirements for a service system. Prod. Oper. Manag. **16**(1), 13–39 (2007)
17. J. Harrison, A. Zeevi, A method for staffing large call centers based on stochastic fluid models. Manuf. Serv. Oper. Manag. **7**(1), 20–36 (2005)

18. A. Ingolfsson, E. Akhmetshina, S. Budge, Y. Li, X. Wu, A survey and experimental comparison of service-level-approximation methods for nonstationary $M(t)/M/s(t)$ queueing systems with exhaustive discipline. INFORMS J. Comput. **19**(2), 201–214 (2007)
19. T. Jiménez, G.M. Koole, Scaling and comparison of fluid limits of queues applied to call centers with time-varying parameters. OR Spectr. **26**(3), 413–422 (2004)
20. G. Jongbloed, G.M. Koole, Managing uncertainty in call centers using Poisson mixtures. Appl. Stoch. Model. Bus. Ind. **17**(4), 307–318 (2001)
21. S. Liao, G.M. Koole, C. van Delft, O. Jouini, Staffing a call center with uncertain non-stationary arrival rate and flexibility. OR Spectr. **34**, 1–31 (2012)
22. V. Mehrotra, O. Ozlük, R. Saltzman, Intelligent procedures for intra-day updating of call center agent schedules. Prod. Oper. Manag. **19**(3), 353–367 (2010)
23. E.J. Pinker, R.C. Larson, Optimizing the use of contingent labor when demand is uncertain. Eur. J. Oper. Res. **144**(1), 39–55 (2003)
24. T. Robbins, Managing service capacity under uncertainty. Ph.D. Thesis, Penn State University, 2007
25. A. Roubos, G.M. Koole, R. Stolletz, Service level variability of inbound call centers. Manuf. Serv. Oper. Manag. **14**(3), 402–413 (2012)
26. H. Shen, J.Z. Huang, Forecasting time series of inhomogeneous Poisson processes with application to call center workforce management. Ann. Appl. Stat. **2**(2), 601–623 (2008)
27. H. Shen, J.Z. Huang, Interday forecasting and intraday updating of call center arrivals. Manuf. Serv. Oper. Manag. **10**(3), 391–410 (2008)
28. S.G. Steckley, S.G. Henderson, V. Mehrotra, Service system planning in the presence of a random arrival rate. Working Paper, 2004
29. J.W. Taylor, A comparison of univariate time series methods for forecasting intraday arrivals at call a center. Manag. Sci. **54**(2), 253–265 (2008)
30. J. Weinberg, L.D. Brown, J.R. Stroud, Bayesian forecasting of an inhomogeneous Poisson process with applications to call center data. J. Am. Stat. Assoc. **102**(480), 1186–1199 (2007)
31. W. Whitt, Staffing a call center with uncertain arrival rate and absenteeism. Prod. Oper. Manag. **15**(1), 88–102 (2006)
32. J. Yoo, Queueing models for staffing service operations. Ph.D. Thesis, University of Maryland, 1996

Chapter 20
MDP for Query-Based Wireless Sensor Networks

Mihaela Mitici

Abstract Increased sensors availability and growing interest in sensor monitoring has lead to an significant increase in the number of sensor networks deployed in the last decade. Simultaneously, the amount of sensed data and the number of queries calling this data significantly increased. The challenge is to respond to the queries in a timely manner and with relevant data, without having to resort to hardware updates or duplication. In this chapter we focus on the trade-off between the response time of queries and the freshness of the data provided. Query response time is a significant Quality of Service for sensor networks, especially in the case of real-time applications. Data freshness ensures that queries are answered with relevant data, that closely characterizes the monitored area. To model the trade-off between the two metrics, we propose a continuous-time Markov decision process with a drift, which assigns queries for processing either to a sensor network, where queries *wait* to be processed, or to a central database, which provides stored and possibly *outdated* data. To compute an optimal query assignment policy, a discrete-time discrete-state Markov decision process, shown to be stochastically equivalent to the initial continuous-time process, is formulated. This approach provides a theoretical support for the design and implementation of WSN applications, while ensuring a close-to-optimum performance of the system.

M. Mitici (✉)
Faculty of Aerospace Engineering, Air Transport and Operations,
Delft University of Technology, Delft, The Netherlands
e-mail: m.a.mitici@tudelft.nl

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_20

20.1 Problem Description

Following increasing computing capabilities of modern sensors, wireless sensor networks (WSNs) have become an integrated platform where local query processing is performed. Thus, not only storage facilities, such as central databases (DB), are able to answer queries, but also the sensors within the WSN. Calls to the WSN are slow and can overload the network, leading to high query response times. But the WSN answers queries with fresh, recently sensed data. Calls to the DB are fast, but the stored data is possibly outdated. A trade-off arises between the response time of queries and the freshness of the data provided.

We analyze the cost of query processing seen as a trade-off between two QoS requirements: (1) the response time of queries, which is the time between a query is initiated until this query is solved and (2) the freshness (age) of the data provided, which is the time between the moment sensed data is generated by a sensor and the time this data is provided to a query (Fig. 20.1).

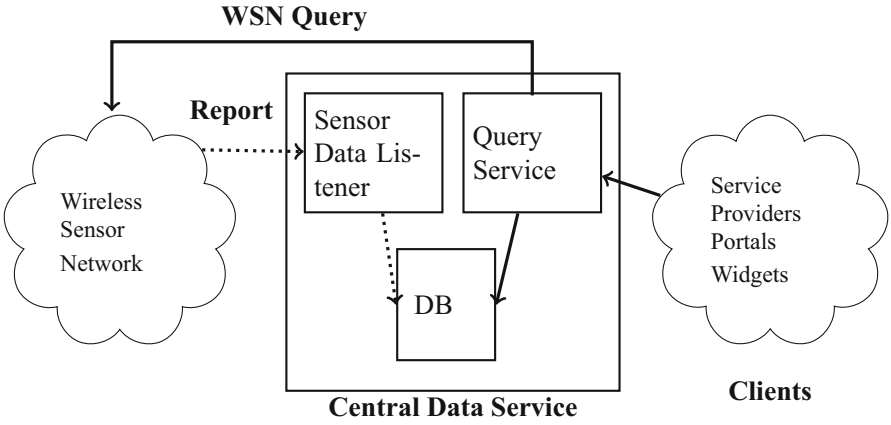


Fig. 20.1: High-level architecture of the query-based system: Queries generated by clients are processed either by the WSN or by the DB. Reports are periodically generated by the WSN and processed only by the WSN. After being processed, reports update the DB

Queries are processed as follows. When a WSN query is initiated, the Query Service of the network requests a sensor value from the network, and upon receiving this request, the network sends its most recent value. We assume a processor sharing service type for the WSN. This reflects the IEEE 802.15.4 MAC design principle of distributing the processing capacity fairly among the jobs present in the network. When a DB query is initiated, the query is answered with DB data, which was stored at a previous time. Data is stored upon reporting. Data reports are periodically pushed to a sink from the sensor network. The sink ultimately updates the DB with the reported values.

A WSN call increases the load of the network and results in possibly large query response times. However, WSN calls are energy efficient. This is because only those values that are of interest are transmitted. In the case of DB calls, the query response time is significantly lower since the data is already stored and available at the DB. However, data reporting increases energy consumption, especially when the reported data is of no (immediate) interest. Also, available DB data is possibly outdated and might no longer accurately reflect the state of the monitored environment.

We model the trade-off between the query response time and data freshness by means of a continuous-time Markov Decision Process with a drift [6]. The continuous character of the process, and in particular, the fact that a state-component of the process evolves continuously over time, makes the problem non-standard and computationally intractable, i.e. the standard way of deriving an optimal policy recursively using dynamic programming is not directly applicable. For computational reasons, we first propose a non-standard exponentially uniformized Markov decision process, which we show to be *stochastically equivalent* to the original continuous-time Markov decision process with a drift. However, the exponentially uniformized process still contains a continuous-state component. For further computational tractability, we next argue a discrete-time and discrete-state Markov decision process. We then determine an optimal query assignment policy for the discrete-time and state process by means of stochastic dynamic programming. Finally, we show that our approach can be used for any given, fixed policy. We consider two query assignment policies, commonly used in practice, and compare their performance with the optimal policy.

20.2 Model Formulation

The system consists of a service facility (WSN) with Processor Sharing capabilities and a storage facility (DB). Figure 20.2 shows the proposed model. Two types of jobs: queries and reports, arrive at the system according to a Poisson process. Queries arrive at rate λ_1. Reports arrive at rate λ_2. Reports are requests issued to the WSN to sense the environment and send the data to the DB for storage. The service requirements of the jobs are exponentially distributed with parameter μ, independently of the job type. To ensure that the system is stable, we assume that $\lambda_2 < \mu$. Queries are handled by a controller which assigns them either to the DB or to the WSN. When assigned to the DB, queries are immediately answered with stored data. However, the stored data might be outdated, i.e., the age of the data might exceed a validity threshold T. Assigned to the WSN, the queries wait to receive the sensed data, sharing the service with the other jobs present in the network. We assume a query processing cost which is influenced by the type of query assignment (DB or WSN assignment). The cost of a DB assignment is an instantaneous cost, indicating how much the age of the stored data has exceeded a validity threshold T. The cost of a WSN assignment consists of penalties, accumulating over time, related to having queries waiting in the WSN to be processed. These penalties increase upon a WSN assignment, as a consequence of the Processor Sharing service type of the

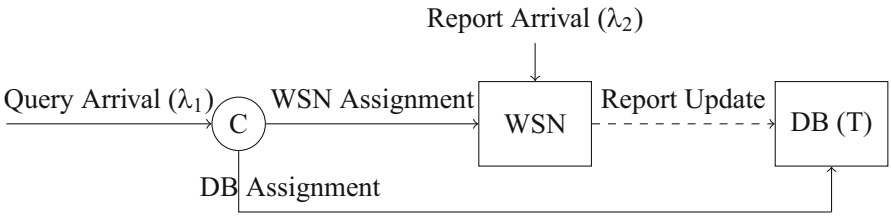


Fig. 20.2: A controller (C) assigns queries to a database (DB) or to the sensor network (WSN). The DB solves queries instantaneously. The WSN solves reports and queries. The data stored in the DB is considered outdated if the age of the data exceeds a validity threshold T. After T is exceeded, the age of the data increases linearly until a report completion updates again the DB

WSN. Upon a query arrival at the controller, the model decides between increasing the query processing cost of the system with an instantaneous DB-related cost or with a WSN-related cost.

We are interested in finding an optimal assignment strategy and in quantifying the assignment cost of this optimal assignment strategy.

20.3 Continuous Time Markov Decision Process with a Drift

The system presented in Sect. 20.2 is formally introduced below as a Continuous Time Markov Decision Process with a drift. Firstly, at any point in time, the system is completely described by the number of queries, reports and the age of the data stored in the DB. Thus, the state space of the problem is defined as follows.

- **State space:** $S = \mathbb{N}_0 \times \mathbb{N}_0 \times [0,\infty)$, where $(i,j,t) \in S$ denotes the state with i queries and j reports in the WSN, and the time t since the last report completion (age of the stored data).

Upon a query arrival, the controller assigns the query for processing either to the DB or to the WSN. The action space is, thus, defined as follows.

- **Action**: the controller takes an action a from the action space $A = \{D,W\}$, where $a = D$ denotes a DB assignment and $a = W$ denotes a WSN assignment.

We define a policy π to be a mapping from the state space $S \to A$, which specifies the action $a \in A$ the controller takes when the system is in state $(i,j,t) \in S$ and a query arrival occurs. We make the natural assumption that this policy is right-continuous in the age component t, which allows for threshold-type of assignment policies of the form $t > T$, where T is a threshold.

The system has a state transition upon a query or report arrival, a query or report completion. The rates at which these events happen are as follows.

- **The transition rates**, when in state $(i,j,t) \in S$ and action $a \in A$ is taken:

$$q^a[(i,j,t),(i,j,t)'] = \begin{cases} \lambda_1, & (i,j,t)' = (i+1,j,t), \quad a = W \\ \lambda_1, & (i,j,t)' = (i,j,t), \quad a = D \\ \lambda_2, & (i,j,t)' = (i,j+1,t) \\ \mu\phi_1(i,j), & (i,j,t)' = (i-1,j,t), i > 0 \\ \mu\phi_2(i,j), & (i,j,t)' = (i,j-1,0), j > 0 \end{cases} \tag{20.1}$$

with $\phi_1(i,j) = \frac{i}{i+j}, \phi_2(i,j) = \frac{j}{i+j}$ indicating the Processor Sharing service discipline assumed for the WSN. The first line of (20.1) models a query arrival under action $a = W$, i.e. the query is assigned to the WSN for processing. The state space illustrates an increment in the number of queries from i to $i+1$. The second line of (20.1) models a query arrival under action $a = D$, i.e. the query is assigned to the DB. In this case, the query is processed immediately, no changes occur in the number of the queries and reports in the system. The third line of (20.1) models a report arrival. The state of the system illustrates an increment in the number of reports. The fourth line of (20.1) models a query completion at the Processor Sharing rate $\phi_1(i,j) = \frac{i}{i+j}$. The number of queries in the system is decremented to $i-1$. Lastly, the fifth line of (20.1) models a report completion at the Processor Sharing rate $\phi_2(i,j) = \frac{j}{i+j}$. The age of the stored data is reset to zero as a report is completed and updates the DB with the most recently sensed data.

The above Markov Decision Process has a deterministic drift for the age component, t. This increases linearly as long as no report is completed. Also, we consider two types of decisions. Firstly, the decision to assign an incoming query to the DB affects only the infinitesimal generator of the Continuous Time Markov Decision Process (see second line of (20.1)). Secondly, the decision to assign a query to the WSN affects both the infinitesimal generator and determines a change in the state of the system (see first line of (20.1)).

The dynamics of this controlled Markovian decision process are uniquely determined by its infinitesimal generators [1]. In the case of our system, under action a, the generator is specified, for any function $f : S \times S \times (0,\infty) \to \mathbb{R}$, as follows:

$$\mathcal{H}^a f(i,j,t) = \sum_{(i,j,t)'} q^a[(i,j,t),(i,j,t)'] \cdot f[(i,j,t)'] + \frac{d}{dt} f(i,j,t) \tag{20.2}$$

The generator in (20.2) shows that, over time: (1) a jump to a new state $(i,j,t)'$ occurs at rate q^d and the time increases or (2) no jump occurs and the time increases.

The cost of the system is two-fold. Firstly, we consider the cost i of having i queries waiting within the WSN to be processed. This cost gives an overview of the load of the WSN. Secondly, we consider an instantaneous cost incurred every time a query is solved by the DB. This is a penalty for each time unit the age of the stored data exceeds a given threshold T. The two costs illustrate the trade-off between the response time of the queries within the WSN and the age of the data provided. Formally, we consider the following cost.

- **Cost**: when in state (i,j,t), a cost rate i for the queries waiting in the WSN and an instantaneous cost $(t-T)^+$, where $x^+ = \max(x,0)$, upon a DB assignment.

20.4 Exponentially Uniformized Markov Decision Process

The continuous character of the process described in Sect. 20.3, and in particular, the continuous age component of the process, which evolves over time, make the query assignment problem computationally intractable, i.e. the standard way of deriving an optimal policy recursively using dynamic programming is not applicable for a Continuous Time Markov Decision Process with a drift. More precisely, the method of uniformization, commonly used to make a Continuous Time Markov Decision Process computationally tractable, is not directly applicable due to the drift (the age component evolving over time) of our process. Uniformization, as introduced in [5], is a well-known technique used to transform a continuous time Markov jump process into a discrete time Markov process. When the state is also discrete, the process is referred to as a continuous time Markov chain (see, for instance, [2, 9]).

In [4] and [10], time discretization is applied to continuous-time Markov decision processes with a drift component evolving over time. Time discretization is a somewhat similar method to uniformization. Time discretization, however, is an approximative method which leads to technical weak convergence. Moreover, it does not lead to exact computational results. To be able to compute an optimal query assignment policy, we construct an exponentially uniformized Markov Decision Process, and show it to be stochastically equivalent to the initial continuous-time Markov decision process with a drift. This implies that the two processes are the same in terms of expected assignment costs and policies. We next construct a discrete time and state Markov decision process, which is computationally tractable. Based on this process, an optimal assignment policy is computed. We then argue and numerically show that the assignment policy computed also holds for the initial continuous-time Markov decision process with a drift (Sect. 20.3).

We now uniformize the continuous-time Markov decision process with a drift described in Sect. 20.3. First, let B be an arbitrarily large finite number such that $B \geq \lambda_1 + \lambda_2 + \mu$. Next, we construct a process which, at exponential times with parameter B, will have a transition from state $(i,j,t) \in S$, as specified in Sect. 20.3, to $(i,j,t)' \in S$. Denote by s the exponential realization time of this transition. Then, given a transition realization of duration s, the transition probabilities under action $a \in A$, from one transition epoch to the next, become:

$$P^a[(i,j,t),(i,j,t)'] = \begin{cases} \lambda_1 B^{-1}, & (i,j,t)' = (i+1,j,t+s), a = W \\ \lambda_1 B^{-1}, & (i,j,t)' = (i,j,t+s), \quad a = D \\ \lambda_2 B^{-1}, & (i,j,t)' = (i,j+1,t+s) \\ \mu B^{-1} \phi_1(i,j), & (i,j,t)' = (i-1,j,t+s), i > 0 \\ \mu B^{-1} \phi_2(i,j), & (i,j,t)' = (i,j-1,0), j > 0 \\ 1 - (\lambda_1 + \lambda_2 + \mu \mathbf{1}_{i+j>0}) B^{-1}, & (i,j,t)' = (i,j,t+s) \\ 0, & \text{otherwise.} \end{cases}$$

Theorem 20.1. *For any policy π, the exponentially uniformized Markov Decision Process and the original Continuous Time Markov Decision Process with a drift are stochastically equivalent.*

Proof. Appendix

A consequence of Theorem 20.1 is that the expected assignment cost for the exponentially uniformized MDP and the CTMDP with a drift are the same. This, in turn, leads to the same optimal policy for the two processes

Now observe that in the CTMDP with a drift, the actions are only taken upon query arrivals, which occur at exponential times. In the case of the exponentially uniformized MDP, the exponential times have parameter B. Thus, the actions will still be taken at exponential times with parameter B, upon a query arrival. Therefore, it is sufficient to keep track of the number of exponential phases N (Erlang distribution with parameter B and N phases). This allows us to restrict ourselves to a discrete-time and space Markov decision process in Sect. 20.5.

20.5 Discrete Time and Discrete Space Markov Decision Problem

Based on the exponentially uniformized model in Sect. 20.4, we formulate our assignment problem as a discrete-time and space Markov decision problem as follows.

- **State space**: $S = \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0$, where $(i, j, N) \in S$ denotes a state with i queries and j reports at the WSN and N is the age of the stored data, with N the number of steps (exponentially distributed with uniformization parameter B) since the last report completion.
- **Action space**: Upon a query arrival, an action a is taken from the action space $A = \{D, W\}$, where $a = D$ is a DB assignment and $a = W$ is a WSN assignment.
- **Transition probabilities**, when in state $(i, j, N) \in S$ and action $a \in A$:

$$P^a[(i,j,N),(i,j,N)'] = \begin{cases} \lambda_1', & (i,j,N)' = (i+1,j,N+1), a = W \\ \lambda_1', & (i,j,N)' = (i,j,N+1), \quad a = D \\ \lambda_2', & (i,j,N)' = (i,j+1,N+1) \\ \mu' \phi_1(i,j), & (i,j,N)' = (i-1,j,N+1), i > 0 \\ \mu' \phi_2(i,j), & (i,j,N)' = (i,j-1,0), j > 0 \\ 1 - (\lambda_1' + \lambda_2' + \mu' \mathbf{1}_{i+j>0}), & (i,j,N)' = (i,j,N+1) \\ 0, & \text{otherwise} \end{cases} \tag{20.3}$$

with $\phi_1(i,j) = \frac{i}{i+j}$, $\phi_2(i,j) = \frac{j}{i+j}$ and $\lambda_i' = \lambda_i B^{-1}, i \in \{1,2\}$ and $\mu' = \mu B^{-1}$ as per uniformization (see Sect. 20.4). The first two lines of (20.3) model query arrivals under action a. The third line of (20.3) models report arrivals. The fourth and fifth

lines of (20.3) model query and report completions, respectively. The sixth line of (20.3) is a dummy transition as a result of the uniformization. The last line of (20.3) prohibits any other state transition. Note that every step, the age is incremented, except when a report is completed, i.e. age reset to zero.

- **Cost function**: The cost of the system is two-fold. Firstly, when i queries are waiting to be solved within the WSN, the system incurs a cost per unit of time:

$$i \tag{20.4}$$

This can be interpreted as, each unit of time, the system pays one unit for each waiting query. At the end of a query's service, the system had payed one unit for each unit of time the query was in the system, i.e. the query response time. Secondly, if an incoming query is assigned to the DB, an instantaneous penalty is incurred for exceeding the validity threshold T of the stored data:

$$\max(N' - T)^+, \ (x)^+ = \max\{0, x\}. \tag{20.5}$$

where $N' = N/B$ is the age of the data in time units, i.e. the number of uniformization steps multiplied by the expected length of a step. In this case, the system pays for the time the validity threshold T is exceeded. Considering the cost of having queries waiting in the WSN (20.4) and the instantaneous cost associated with a DB assignment (20.5), when the system is in state (i, j, N), the cost incurred per unit of time is:

$$C^a(i, j, N) = i + \lambda_1 (N' - T)^+ \mathbf{1}_{(a=D)}, \ \text{where} (x)^+ = \max\{0, x\}. \tag{20.6}$$

Equation (20.6) shows that the model assesses the trade-off between increasing the processing cost of the system by the instantaneous cost $(N' - T)$ or by accumulating i units of penalties every time slot, until a change in the number of queries occurs.

Remark 1: The number of exponential phases approximates the time until a report completion by $t + s = (N + 1) \cdot B^{-1}$. Also, the variance of an Erlang distribution with N phases and parameter B, which is the case for our discretized age, is $\frac{N+1}{B^2}$. As $B \geq \lambda_1 + \lambda_2 + \mu$ can be chosen arbitrarily large (see [3]), by the law of large numbers, for very large B, the distribution of Erlang(N+1,B) will concentrate around $(N + 1) \cdot B^{-1}$. Thus, for large uniformization parameter B, the Discrete Time and State MDP approximates the uniformized MDP arbitrarily close.

Remark 2: The value of the uniformization parameter $B \leq \lambda_1 + \lambda_2 + \mu$ can be seen as a scaling factor that does not influence the results. One could expect that, for small values of B, a minor effect on the policy might be present due to the approximation of the age component $N' = N/B$ (see Sect. 20.8.2). However, we have not been able to find any such example. In other words, the approach followed is strongly supported, both theoretically and numerically.

20.6 Standard Markov Decision Process

Based on the model in Sect. 20.5, the quadruple (S,A,P,C) completely describes the discrete-time and state MDP. To determine an optimal assignment policy and to use standard dynamic programming, we define the following value function:

$\mathbf{V}_n(i,j,N) =$ minimal expected assignment cost over n steps starting in state (i,j,N).

Then $\mathbf{V}_n(i,j,N)$ is computed recursively by means of the value iteration algorithm (see, for instance, [8, Sect. 8.5.1]) as follows. First, we consider $\mathbf{V}_0(i,j,N) = 0$. Next, we iterate according to the value iteration algorithm and the following backward recursive equation:

$$\mathbf{V}_{n+1}(i,j,N) = \begin{cases} i' + \lambda_1' \min \begin{cases} V_n(i+1,j,N+1) \\ (N-T')^+ + V_n(i,j,N+1) \end{cases} \\ +\lambda_2' V_n(i,j+1,N+1) \\ +\mu' \phi_1(i,j) V_n(i-1,j,N+1)\mathbf{1}_{i>0} \\ +\mu' \phi_2(i,j) V_n(i,j-1,0)\mathbf{1}_{j>0} \\ +[1-(\lambda_1' + \lambda_2' + \mu' \mathbf{1}_{i+j>0})] V_n(i,j,N+1). \end{cases} \tag{20.7}$$

where $i' = i/B$ and $T' = T/B$, following uniformization. The first term of (20.7) is the cost of having i queries in service and a query assigned to either the WSN or the DB. The next three terms represent the cost incurred by a transition due to a report arrival, a query completion and a report completion, respectively. The final term is the dummy term due to uniformization.

Simultaneously with computing $\mathbf{V}_n(i,j,N)$, the algorithm computes a ε-optimal stationary policy π_n which associates an optimizing action with the right-hand side of (20.7) for any state (i,j,N). Given the assignment policy, it is possible to compute the average assignment cost. Denote the minimal average assignment cost by g^*. Since the underlying Markov chain is ergodic, g^* is independent of the initial state. We approximate g^* using the following bounds developed in [7]:

$$\begin{aligned} L_n^* \le g^* \le L_n^{**}, \text{ where} \\ L_n^* = \min[V_{n+1}(i,j,N) - V_n(i,j,N)], \\ L_n^{**} = \max[V_{n+1}(i,j,N) - V_n(i,j,N)]. \end{aligned} \tag{20.8}$$

In (20.8), L_n^* is the minimum difference of the value function over two iteration steps, n and $n+1$, whereas L_n^{**} is the maximum difference of the value function over steps n and $n+1$. For $n \to \infty$, L_n^* and L_n^{**} become arbitrarily close. The optimal cost g^* is computed with an accuracy ε by iterating the right-hand side of (20.7) for n times until $L_n^{**} - L_n^* \le \varepsilon/B$, with B the uniformization parameter. The average assignment cost is approximated as $g^* = \frac{(L_n^{**} + L_n^*)}{2}$. It can be shown that the lower and upper bound converge in a finite number of steps (Theorem 8.5.4 [8]) to the ε-optimal cost.

20.7 Fixed Assignment Policies

In practice, simple assignment policies are employed to manage the query traffic. Observe that following the procedure presented in Sect. 20.4, we can analyze any fixed query assignment policy. Below we consider two fixed policies, commonly used in practice.

20.7.1 Always Assign Queries to the DB

We consider a fixed assignment strategy π^D that always assigns incoming queries to the DB. Upon a query arrival, the cost incurred is $(N-T)^+$. Then (20.7) becomes:

$$\mathbf{V}_{n+1}^{DB}(j,N) = \begin{cases} \lambda_1'[(N-T')^+ + V_n^{DB}(j,N+1)] \\ +\lambda_2' V_n^{DB}(j+1,N+1) \\ +\mu' V_n^{DB}(j-1,0)\mathbf{1}_{j>0} \\ +[1-(\lambda_1'+\lambda_2'+\mu'\mathbf{1}_{j>0})]V_n^{DB}(j,N+1). \end{cases} \tag{20.9}$$

where $T' = T/B$, following uniformization.

20.7.2 Always Assign Queries to the WSN

We consider a fixed assignment strategy π^W that always assigns incoming queries to the WSN. Then (20.7) becomes:

$$\mathbf{V}_{n+1}^{WSN}(i,j,N) = \begin{cases} i' + \lambda_1' V_n^{WSN}(i+1,j,N+1) \\ +\lambda_2' V_n^{WSN}(i,j+1,N+1) \\ +\mu'\phi_1(i,j)V_n^{WSN}(i-1,j,N+1)\mathbf{1}_{i>0} \\ +\mu'\phi_2(i,j)V_n^{WSN}(i,j-1,0)\mathbf{1}_{j>0} \\ +[1-(\lambda_1'+\lambda_2'+\mu'\mathbf{1}_{i+j>0})]V_n^{WSN}(i,j,N+1). \end{cases} \tag{20.10}$$

where $i' = i/B$ and $T' = T/B$, following uniformization.

20.8 Numerical Results

20.8.1 Performance of Fixed Policies vs. Optimal Policy

Figure 20.3 shows the query assignment costs under the optimal assignment strategy and the fixed strategy π^D, for various tolerance thresholds T. The fixed strategy π^W is independent of T. As expected, as T increases, π^{DB} converges to the optimal assignment policy. This because the stored data is considered fresh for a longer time and thus, it is more often optimal to assign an incoming query to the DB.

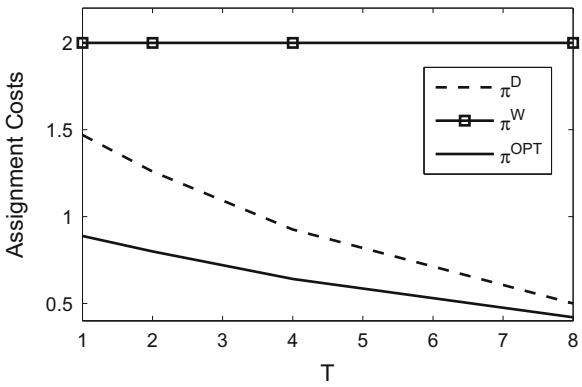


Fig. 20.3: Assignment costs, $\lambda_1 = 0.2, \lambda_2 = 0.1, \mu = 0.4, T \in \{1,2,4,8\}$

20.8.2 Optimal Policy Under Different Values of the Uniformization Parameter

As argued in Sect. 20.5, the uniformization parameter $B \leq \lambda_1 + \lambda_2 + \mu$ can be seen as a scaling factor that does not influence the results. Several numerical examples have been investigated in Fig. 20.4, also showing no effect of B on the assignment policy. Figure 20.4 shows that as the uniformization parameter B increases, the structure of the optimal policy remains the same.

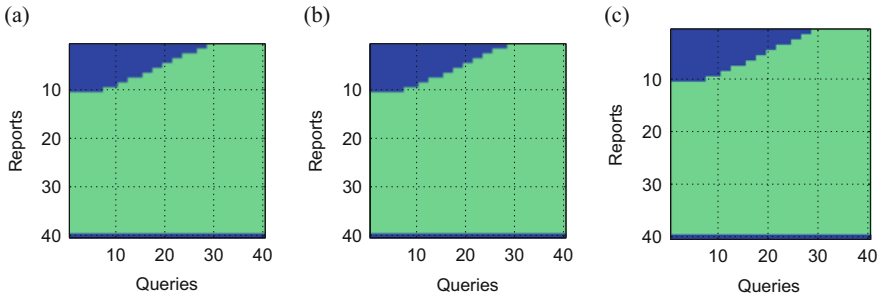


Fig. 20.4: Various uniformization parameter B. WSN (*blue*), DB (*green*) assignment, $T = 1$. (a) $B = \lambda_1 + \lambda_2 + \mu, N = 30$. (b) $B = 5(\lambda_1 + \lambda_2 + \mu), N = 150$. (c) $B = 10(\lambda_1 + \lambda_2 + \mu), N = 300$

20.9 Conclusion

We provided a formal support for the analysis of query processing strategies for wireless sensor networks. We determined, using a Markov decision processes framework, an optimal assignment policy which assigns queries for processing either to the WSN or to a central DB. The optimal policy is based on the trade-off between the penalties related to having queries waiting to be process in the WSN and an instantaneous cost related to the age of the data stored at a central DB. We also argued that our framework can be used for any given, fixed query assignment policy. Lastly, we considered two fixed assignment policies, commonly used in practice, and we compared numerically their performance to the optimal assignment policy.

Appendices

Proof of Theorem 1

Proof (Theorem 20.1). Let $h(x)$ be a measurable function on some state space E of a Markov process. Let $P(v,x,\Xi)$ be a transition function expressing the probability that a process which started in a state x is in the set Ξ at time v. Let $T_v h(x) = \int_E P(v,x,dy)h(y)$ denote a shift operator on the space E. Then the operator

$$\mathcal{H}h(x) = \lim_{v \to 0} \frac{T_v h(x) - h(x)}{v}$$

is called the infinitesimal generator of the Markov process. The quantity $\mathcal{H}h(x)$ can be interpreted as the mean infinitesimal rate of change of the process starting

in state x. Moreover, the infinitesimal generator uniquely define a Markov process [1, Chap. 1]. Therefore, it is sufficient to show that the infinitesimal generator of the exponential uniformized Markov decision process and the original continuous-time Markov decision process with a drift are identical.

In our setting, we consider the state $x = (i, j, t)$. Before addressing the infinitesimal generator of the exponentially uniformized Markov process defined in Sect. 20.4, we first define the transition probability measure under action $a \in A$. Let $\mathbf{P}^a_{\Delta t}$ denote the transition probability measures over a time interval of length $\Delta t > 0$, given that at the last jump the system is in state (i, j, t) and that following a upon a next jump, which occurs in the interval Δt, decision d is taken and the system is in a new state.

As we implicitly made the assumption that a policy π, prescribing an action a upon a query arrival, when the system is in state (i, j, t), is right continuous and since the set of decisions is finite and discrete, for any state (i, j, t) and fixed policy π there exists a $\Delta t > 0$ such that:

$$\pi(i, j, t + u) = \pi(i, j, t) = a, \text{ for all } u \leq \Delta t.$$

Let $f : \mathbb{N} \times \mathbb{N} \times \mathbb{R}$ be an arbitrary real valued function, differentiable in t. Then by conditioning upon the exponential jump epoch with variable χ and for arbitrary function f we obtain,

$$\begin{aligned}
\mathbf{P}^a_{\Delta t} f(i, j, t) &= e^{-\Delta t \cdot \chi} f(i, j, t + \Delta t) \\
&\quad + \int_0^{\Delta t} \chi e^{-u\chi} \sum_{(i,j,t)'} \mathbf{P}^a[(i, j, t), (i', j', t + u)] f(i', j', t + u) du + o(\Delta t)^2 \\
&= f(i, j, t + \Delta t) - \Delta t \chi f(i, j, t + \Delta t) \\
&\quad + \Delta t \chi \sum_{(i',j') \neq (i,j)} q^a[(i, j, t), (i', j', t)] f(i', j', t + \Delta t) \chi^{-1} \\
&\quad + \Delta t \chi [1 - q^a(i, j) \chi^{-1}] f(i, j, t + \Delta t) + o(\Delta t)^2 \\
&= f(i, j, t + \Delta t) + \chi \sum_{(i',j') \neq (i,j)} q^a[(i, j, t), (i', j', t)][f(i', j', t + \Delta t) \\
&\quad - f(i, j, t + \Delta t)] + o(\Delta t)^2,
\end{aligned}$$

where we have used that $q^a[(i, j, t), (i', j', t)] = q^a[(i, j, t + u), (i', j', t + u)]$ for any $(i', j') \neq (i, j)$ and arbitrary s. The term $o(\Delta t)^2$ reflects the probability of at least two jumps and the second term of the Taylor expansion for $e^{-\Delta \chi}$.

Hence, by subtracting $f(i, j, t)$, dividing by Δt and letting $\Delta t \to 0$, we obtain,

$$\begin{aligned}
\frac{\mathbf{P}^a_{\Delta t} f(i, j, t) - f(i, j, t)}{\Delta t} &= [f(i, j, t + \Delta t) - f(i, j, t)]/\Delta t \\
&\quad + \chi [f(i, j, t + \Delta t) - f(i, j, t)] + o(\Delta t)^2 \\
&\quad + \sum_{(i',j') \neq (i,j)} q^a[(i, j, t), (i', j', t)][f(i', j', t) - f(i, j, t)]
\end{aligned}$$

$$\begin{aligned}
&\rightarrow \frac{d}{dt} f(i,j,t) \\
&\quad + \sum_{(i',j') \neq (i,j)} q^a[(i,j,t),(i',j',t)][f(i',j',t) - f(i,j,t)] \\
&\quad = \mathcal{H}^a f(i,j,t), \text{which is the generator in (20.2).}
\end{aligned}$$

Since the exponentially uniformized Markov decision process (defined in Sect. 20.4) and the continuous-time Markov decision process with a drift (defined in Sect. 20.3) share the same generators [1], the two processes are stochastically equivalent.

Notation

S	State space
(i,j,N)	A state, given a discrete state space
$C^a(i,j,N)$	Expected one step cost rate in state (i,j,N), under action a
A	Set of actions available in state (i,j,N)
a	Action when in state (i,j,N), given a discrete state space
W, DB	Stationary Policy
P^W, P^{DB}	One step transition probability distribution/matrix under policy W, DB
$P^a[(i,j,N),(i,j,N)']$	Transition probability into state $(i,j,N)'$, from state (i,j,N), under action a
$q^a[(i,j,N),(i,j,N)']$	Transition rate from state (i,j,N) into $(i,j,N)'$ under action a
$V_n^W(i,j,N), V_n^{DB}(i,j,N)$	Value function under policy W, DB of expected cumulative cost over n steps
$V_n(i,j,N)$	Optimal value function of expected cumulative cost over n steps, starting in state (i,j,N)
$g*$	Optimal average expected cost function
B	Uniformization parameter
$\mathcal{H}$	Infinitesimal generator of Markov decision process

References

1. E.B. Dynkin, *Markov Processes*, vol. 1 (Academic, New York, 1965)
2. I.I. Gikhman, A.V. Skorokhod, *The Theory of Stochastic Processes: II*, vol. 232 (Springer, New York, 2004)

3. A. Hordijk, R. Schassberger, Weak convergence for generalized semi-markov processes. Stoch. Process. Appl. **12**(3), 271–291 (1982)
4. A. Hordijk, F.A. van der Duyn Schouten, Discretization and weak convergence in Markov decision drift processes. Math. Oper. Res. **9**(1), 112–141 (1984)
5. A. Jensen, Markoff chains as an aid in the study of markoff processes. Scand. Actuar. J. **1953**(sup1), 87–91 (1953)
6. M. Mitici, M. Onderwater, M. de Graaf, J.-K. van Ommeren, N. van Dijk, J. Goseling, R.J. Boucherie, Optimal query assignment for wireless sensor networks. AEU-Int. J. Electron. Commun. **69**(8), 1102–1112 (2015)
7. A.R. Odoni, On finding the maximal gain for Markov decision processes. Oper. Res. **17**(5), 857–860 (1969)
8. M.L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (Wiley, New York, 1994)
9. N. van Dijk, On a simple proof of uniformization for continuous and discrete-state continuous-time markov chains. Adv. Appl. Probab. **22**(3), 749–750 (1990)
10. N. van Dijk, A. Hordijk, Time-discretization for controlled Markov processes. I. General approximation results. Kybernetika **32**(1), 1–16 (1996)

Part VI
Financial Modeling

Chapter 21
Optimal Portfolios and Pricing of Financial Derivatives Under Proportional Transaction Costs

Jörn Sass and Manfred Schäl

Abstract A utility optimization problem is studied in discrete time $0 \leq n \leq N$ for a financial market with two assets, bond and stock. These two assets can be traded under transaction costs. A portfolio (Y_n, Z_n) at time n is described by the values Y_n and Z_n of the stock account and the bank account, respectively. The choice of (Y_n, Z_n) is controlled by a policy. Under concavity and homogeneity assumptions on the utility function U, the optimal policy has a simple cone structure. The final portfolio (Y_N^*, Z_N^*) under the optimal policy has an important property. It can be used for the construction of a consistent price system for the underlying financial market.

Key words: Numeraire portfolio, Utility function, Consistent price system, Proportional transaction costs, Dynamic programming

21.1 Introduction

We will start with discrete-time utility optimization which is now a classical subject and can be treated as a Markov decision process in discrete time $0 \leq n \leq N$. Our main goal will be an application to adequate pricing of financial derivatives, in particular options, which is an important subject of financial mathematics. A financial market is studied where two assets, bond and stock, can be traded under transaction

J. Sass (✉)
Fachbereich Mathematik, TU Kaiserslautern, Erwin-Schrödinger-Str.,
Kaiserslautern D-67663, Germany
e-mail: sass@mathematik.uni-kl.de

M. Schäl
Institut für Angewandte Mathematik, Universität Bonn, Endenicher Allee 60,
Bonn D-53115, Germany
e-mail: schael@iam.uni-bonn.de

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_21

costs. A mutual fund is a good example for the stock. Under concavity and homogeneity assumptions on the utility function U, it is known that the optimal policy has a cone structure not only for models without but also for models with linear transaction costs, see below. In the present paper we will focus on such models.

An Explanatory Model

In order to describe the application of the optimal policy from utility maximization to pricing of financial derivatives, let us first consider a simple model with only one period $[0,N]$ (starting in 0 and finishing in $N=1$) and without transaction costs. Let B_N be the value on the bank account at N if we start with one unit of money $B_0=1$. Then B_N^{-1} is the classical discount factor. For fixed initial wealth x, the policy can be described by a real number θ, the investment in the stock. Then the wealth at N is $X_N^\theta=(x-\theta)B_N+S_0^{-1}\theta S_N=B_N(x-\theta+S_0^{-1}\theta B_N^{-1}S_N)$, where S_0 and S_N are the stock prices at 0 and N and $S_0^{-1}\theta$ is the invested number of stocks.

The classical present value principle for pricing future incomes is based on the expectation of discounted quantities. According to this principle, an adequate price for a contingent claim offering S_N, i.e. one unit of stock, at N would be $\mathrm{pr}(S_N)=E[B_N^{-1}S_N]$. But this answer may be wrong, because we know in the present situation of a financial market that S_0 is the adequate price. Starting with S_0 one is sure to have S_N at N. But in general one has $E[B_N^{-1}S_N]\neq B_0^{-1}S_0=S_0$ and not the equality one would like to have. Note that the equality means that the discounted stock price process $\{B_0^{-1}S_0,B_N^{-1}S_N\}$ is a martingale. It was a great discovery for the stochastic community when one realized that martingales come into play. This is the reason for a change of measure where the original real-world probability measure P is replaced by an artificial martingale measure Q with Radon-Nikodym density q w.r.t. P. One wants to study adequate prices $\mathrm{pr}(C)$ for a contingent claim C depending on the underlying financial derivative and maturing at N. In the present simple model, one has $C=f(S_N)$ for some function f, since S_N is the only random variable. In multiperiod models, C is contingent upon the whole development of the stock up to N. After a change of measure, one considers the present value principle under Q:

$$\mathrm{pr}(C)=E[qB_N^{-1}C]=E_Q[B_N^{-1}C] \quad \text{with} \quad \mathrm{pr}(S_N)=E_Q[B_N^{-1}S_N]=S_0. \tag{21.1}$$

Then $\{B_0^{-1}S_0,B_N^{-1}S_N\}$ is a martingale under Q and $\mathrm{pr}(\cdot)$ is called a consistent price system because of the relation $\mathrm{pr}(S_N)=S_0$. In general however, one has several choices for a martingale measure Q and one has to specify an additional preference in order to distinguish one measure Q and thus one generally agreed prize. Therefore, no preference-independent pricing of financial derivatives is possible.

Construction of a Price System

Now we explain the relations to utility optimization and how to construct a martingale measure Q and thus a consistent price system by the optimal investment θ^*. Let us consider the portfolio optimization problem where the wealth at N is $X_N^\theta = B_N(x-\theta+S_0^{-1}\theta B_N^{-1}S_N)$ defined as above and where we study $\max_\theta E[U(B_N^{-1}X_N^\theta)]$. Then we get for the optimal investment θ^* by differentiating:

$$E\left[U'\left(B_N^{-1}X_N^{\theta^*}\right)\left(S_0^{-1}B_N^{-1}S_N-1\right)\right]=0 \quad \text{or} \quad E\left[cU'(B_N^{-1}X_N^{\theta^*})B_N^{-1}S_N\right]=S_0,$$

if the constant c is chosen such that $E[cU'(B_N^{-1}X_N^{-1})]=1$. By a simple calculation one obtains $c=xE[U^*(B_N^{-1}X_N^{\theta^*})]^{-1}$ with $U^*(w):=U'(w)w$. Now we can set $q=cU'(B_N^{-1}X_N^{\theta^*})$ for q as above and we get

$$\mathrm{pr}(C)=xE\left[U^*\left(B_N^{-1}X_N^{\theta^*}\right)\right]^{-1}E\left[U'\left(B_N^{-1}X_N^{\theta^*}\right)B_N^{-1}C\right] \tag{21.2}$$

where typically $x=1$. In fact we then have $E[qB_N^{-1}S_N]=S_0$ and q thus defines a martingale measure. By a 'marginal rate of substitution' argument it can be shown how this price depends in a traditional way on the investor's preference or relative risk aversion (see Davis [7], Schäl [26, Introduction]).

The Numeraire Portfolio

In the present paper, a special martingale measure Q is studied which is defined by the concept of the *numeraire portfolio*. Then the choice of Q can be justified by a change of numeraire (discount factor) in place of a change of measure. For this approach one has to choose for U the log-utility with $U'(w)=w^{-1}$ and $U^*(w)=1$ (see Becherer [2], Bühlmann and Platen [3], Christensen and Larsen [4], Goll and Kallsen [9], Karatzas and Kardaras [13], Korn et al. [17], Korn and Schäl [15, 16], Long [19], Platen [21], Schäl [25]). The optimal investment θ^* is called log-optimal. In fact, then one obtains $q=c\,(B_N^{-1}X_N^{\theta^*})^{-1}$ and $\mathrm{pr}(C)=E[qB_N^{-1}C]=E[c\,(X_N^{\theta^*})^{-1}C]$ and $c=1$ for $x=1$ since $U^*(w)=1$. As a result we finally get

$$\mathrm{pr}(C)=E[(X_N^{\theta^*})^{-1}C]. \tag{21.3}$$

Comparing (21.1) with the possibly wrong prize $\mathrm{pr}(C)=E[B_N^{-1}C]$ (see above) and with a consistent prize (21.1), we see the following: In (21.3) we stick to the original probability measure but replace B_N with the wealth $X_N^{\theta^*}$ which can be realized on the market when starting with $x=1$ on the bank account and investing according to θ^*. When looking for a discount factor, we thus assume that we will use $x=1$ in an optimal way instead of investing exclusively in the bank account. By the way, as a consequence the (generalized) discount factor $(X_N^{\theta^*})^{-1}$ is random.

We think that it is easier to explain a change of the discount factor to a non-expert than a change of measure since we here have a financial market where we have more choices for investing one unit of money and not only the choice to invest in the bank account.

The General Model with Transaction Costs

The problem of the paper is to carry over this idea to multiperiod financial models (where $N \geq 1$) in the presence of transaction costs. For such models, utility maximization and in particular log-optimality are also well studied. The wealth at stage n will be given by portfolios (Y_n, Z_n) with generic values (y,z) describing the value of the stock account and the bank account at time n, respectively. It is known that the log-optimal dynamic portfolio can be described by two *Merton lines* in the (y,z)-plane (see Kamin [12], Constantinides [5], Sass [22]) in place of one Merton line as in the setting without transaction costs. For results in continuous time see Davis and Norman [8], Magill and Constantinides [20] and Shreve and Soner [27].

Here we will contribute to that theory. We need a natural region for portfolios (y,z) and therefore allow for negative values of y and z (but with $y+z>0$), i.e. for short selling and borrowing. For any stage $n<N$, the region of admissible portfolios will be the *solvency region* and it is divided by the two Merton lines into three cones where it is optimal either (i) to buy (ii) to sell or (iii) not to trade, respectively. These properties simplify numerical studies considerably. When looking for a natural region, 'natural' means that it is as large as possible and that these three cones are not empty. The latter fact can happen if one restricts to nonnegative values of y and z. We will provide a moment condition (R3) on the returns for the latter property. Furthermore we will deal with open action spaces in order to be sure that the optimal action lies in the interior. This is needed for the argument that the derivative vanishes at a maximum point which was also used above in the simple explanatory model.

Martingale Measures and the Numeraire Portfolio

Martingale measures and price systems are also discussed in the literature for models with transaction costs, see Jouini and Kallal [10], Koehl et al. [14], Kusuoka [18], Schachermayer [24]. As explained above, they are basic for the concept of a numeraire portfolio. Now the goal of the paper is the following: Study the log-optimal dynamic portfolio and show that it defines a numeraire portfolio. The definition of martingale measures is not so evident in the presence of transaction cost.

When maximizing the expected utility $E[U(B_N^{-1}(Y_N+Z_N))]$, we will use Y_N+Z_N as total wealth at time N as in Bäuerle and Rieder [1, Sect. 4.5] and Cvitanić and Karatzas [6]. A more general concept can also be used where one introduces

liquidation costs L at time N and considers $L(Y_N)+Z_N$ in place of Y_N+Z_N . For this problem we refer the reader to Sass and Schäl [23]. Since L is not differentiable, this case would cause a lot of additional problems and additional assumptions are needed. Indeed, this paper aims at providing the proof in the case without liquidation costs, since this case allows for much more straightforward arguments and requires less assumptions.

A contingent claim C, maturing in N, is split into a contingent claim Y^C for the stock account and a contingent claim Z^C for the bank account. Then a price for (Y^C,Z^C) turns out to be

$$\mathrm{pr}(Y^C,Z^C)=E\left[(Y_N^*+Z_N^*)^{-1}(Y^C+Z^C)\right]. \tag{21.4}$$

Here $Y_N^*+Z_N^*$ is the wealth at N under the optimal dynamic portfolio. The role of $(Y_N^*+Z_N^*)^{-1}$ is that of a generalized discount factor and (Y_N^*,Z_N^*) is then called a numeraire portfolio at N.

Main Result

As main result, the log-optimal portfolio indeed turns out to define a numeraire portfolio also for models with transaction costs. As in the classical case without transactions costs, the message is the following: under very general conditions you don't need to change the measure for pricing a contingent claim. You can stick to the probability measure P describing the real market and thus being open to statistical procedures. Instead of the bank account you must use the wealth of the log-optimal policy, starting with one unit of money as usual, as reference unit or benchmark (in the terminology of Platen [21]). Thus we see a contingent claim C relative to $Y_N^*+Z_N^*$. Working with P is also extremely useful when integrating the modeling of risk into finance as in combined finance and insurance problems, see Bühlmann and Platen [3].

21.2 The Financial Model

The *bond* with prices B_n, $n=0,\ldots,N$, will be described by positive deterministic *interest rates* $r_n-1\geq 0$ and the *stock* with prices S_n, $n=0,\ldots,N$, will be described by the *relative return process* consisting of positive independent random variables $\{R_n-1,n=1,\ldots,N\}$. Let $B_0=1$ and $S_0>0$ be deterministic. Then

$$B_n=B_{n-1}r_n,\quad B_n^{-1}S_n=B_{n-1}^{-1}S_{n-1}R_n,\quad n=1,\ldots,N. \tag{21.5}$$

We write $\mathbb{F}=\{\mathcal{F}_n,n=0,\ldots,N\}$ for the filtration generated by $\{R_n,n=1,\ldots,N\}$ where $\mathcal{F}_0$ is trivial and $\mathcal{F}=\mathcal{F}_N$.

A *trading strategy* is given by a real valued $\mathbb{F}$-adapted stochastic process $\{\Delta_n, 0 \leq n < N\}$ describing the amount of money (wealth) invested in the stock. For the transaction Δ_n, the total cost $K(\Delta_n)$ with *transaction costs* $0 \leq \mu < 1$, $\lambda \geq 0$ has to be paid, where

$$K(\theta) := (1+\lambda)\theta \text{ for } \theta \geq 0, \quad K(\theta) := (1-\mu)\theta \text{ for } \theta \leq 0. \tag{21.6}$$

A trading strategy will define a *dynamic portfolio* $\{(Y_n, Z_n), 0 \leq n \leq N\}$ describing the wealth $\{Y_n\}$ on the stock account and the wealth $\{Z_n\}$ on the bank account. We get the *budget equations*

$$Y_n = \overline{Y}_{n-1} r_n R_n, \quad Z_n = \overline{Z}_{n-1} r_n \tag{21.7}$$

$$\overline{Y}_{n-1} = Y_{n-1} + \Delta_{n-1}, \quad \overline{Z}_{n-1} = Z_{n-1} - K(\Delta_{n-1}), \tag{21.8}$$

where $\overline{Y}_{n-1}$ and $\overline{Z}_{n-1}$ are the wealth on the stock account and the bank account after trading. We consider *self-financing* trading strategies where no additional wealth is added or consumed. Then we have $K(y) \geq y$ and $K(\alpha y) = \alpha K(y)$ (positive homogeneity).

We will only consider *admissible* trading strategies where the investor stays solvent at any time in the following sense:

$$(a) \quad Y_N + Z_N > 0 \quad \text{and} \quad (b) \quad Z_n - K(-Y_n) > 0 \quad \text{for} \quad n < N. \tag{21.9}$$

Note that (21.9) implies $Y_n + Z_n > 0$ for $n \leq N$.

21.3 The Markov Decision Model

To ease notation we shall now assume $r_n = 1$ and thus $B_n = 1$, $1 \leq n \leq N$. This a usual assumption and means that one uses directly discounted quantities as $B_n^{-1} S_n$ and $B_n^{-1} B_n = 1$ instead of S_n and B_n.

We will work with a Markov decision process where the state is described by (y,z) where y denotes the wealth on the stock account and z the wealth on the bank account.

Definition 21.3.1.

a. The *state space* at n is $\mathcal{S}_N := \{(y,z) : y+z > 0\}$ for $n = N$ and $\mathcal{S} := \{(y,z) : z - K(-y) > 0\} = \{(y,z) : (1-\mu)y + z > 0, (1+\lambda)y + z > 0\}$ for $n < N$.
b. An *action* θ will denote the transaction describing the amount of money (wealth) invested in the stock. The set of admissible actions will be defined below.
c. The *law of motion* is defined by the budget Eqs. (21.7) and (21.8) where $\{R_n, n = 1, \ldots, N\}$ are independent (but not necessarily identically distributed) random variables. Thus, given the state (y,z) and the action θ at $n-1$, the distribution of the state at n is that of

$$((y+\theta)R_n, z - K(\theta)).$$

$\mathcal{S}_N$ is called the *solvency region* at stage N and $\mathcal{S}$ is called the *solvency region* at all stages $n < N$. Obviously $\mathcal{S}_N$ is defined as $\mathcal{S}$ replacing (λ, μ) by $(0,0)$. Thus, $\mathcal{S}_N$ and $\mathcal{S}$ are open convex cones and the boundaries are formed by half-lines. The condition (21.9) can be written as $(Y_N, Z_N) \in \mathcal{S}_N$ and $(Y_n, Z_n) \in \mathcal{S}$ for $n < N$. We will make the following assumptions on R_n.

Assumption 21.3.2. We assume for $n = 1, \ldots, N$ that R_n is bounded by real constants $\underline{R}, \overline{R}$ with

$$\begin{aligned} &(\mathrm{R1}) \quad 0 < \underline{R} \le R_n \le \overline{R}, \\ &(\mathrm{R2}) \quad \underline{R} < 1 - \mu, \quad 1 + \lambda < \overline{R}, \\ &(\mathrm{R3}) \quad E[(R_n - \underline{R})^{-1}] = E[(\overline{R} - R_n)^{-1}] = \infty. \end{aligned}$$

For convenience, we omit the index n for $\underline{R}, \overline{R}$. Assumption (R3) implies that $\underline{R}$, $\overline{R}$ are in the support of R_n. Then (R2) implies a no-arbitrage condition, i.e., there is a chance that one can loose money and that one can win money when investing in the stock. Assumption (R3) is by far not necessary. Indeed, one only needs that $E[(R_n - \underline{R})^{-1}]$ and $E[(\overline{R} - R_n)^{-1}]$ are big enough. But it is complicated to quantify this property for each stage. Assumption (R3) is satisfied if $P(R_n = r) > 0$ for $r = \underline{R}$, $\overline{R}$ or if R_n has the uniform distribution on $[\underline{R}, \overline{R}]$.

Definition 21.3.3. $\Gamma := \{(y,z) : (yr, z) \in \mathcal{S} \text{ for } \underline{R} \le r \le \overline{R}\}$ and Γ_N are the *presolvency regions* where Γ_N is defined as Γ replacing $\mathcal{S}$ with $\mathcal{S}_N$ and thus (λ, μ) by $(0,0)$.

Obviously Γ_N contains all states at time $N-1$ after trading such that the system is in $\mathcal{S}_N$ at time N for every possible value r of R_N. Assumption (R2) now guarantees that $\Gamma_N \subset \mathcal{S}$ and one can move from any state $(y,z) \in \mathcal{S} \setminus \Gamma_N$ to a state $(y + \theta, z - K(\theta)) \in \Gamma_N$ by buying $(\theta > 0)$ or selling $(\theta < 0)$.

Lemma 21.3.4. $\Gamma = \{(y,z) : (1-\mu)\underline{R}y + z > 0, (1+\lambda)\overline{R}y + z > 0\}$ *and* $\Gamma_N = \{(y,z) : \underline{R}y + z > 0, \overline{R}y + z > 0\}$. Γ *and* Γ_N *are closed convex cones and their boundaries are formed by two rays.*

Definition 21.3.5. The *set of admissible actions* θ at stage $n < N-1$ will be chosen as

$$\mathcal{A}(y,z) := \{\theta : (y + \theta, z - K(\theta)) \in \Gamma\}, \quad (y,z) \in \mathcal{S},$$

and at stage $N-1$ as $\mathcal{A}_{N-1}(y,z)$ defined as $\mathcal{A}(y,z)$ replacing Γ with Γ_N.

Thus $\Delta_{n-1} \in \mathcal{A}(Y_{n-1}, Z_{n-1})$ implies $(Y_n, Z_n) \in \mathcal{S}$ for $n < N$. Important quantities will depend on the state (y,z) only through $y/(y+z)$ and are thus independent of α on the ray $\{(\alpha y, \alpha z) : \alpha > 0\}$. This fact will entail an important cone structure. Therefore we introduce the *risky fraction*

$$\Pi_n := Y_n/(Y_n + Z_n). \tag{21.10}$$

We will restrict attention to situations where $Y_n + Z_n$ is strictly positive. Then Π_n is well-defined.

Convention 21.3.6. If y, z, and π appear in the same context, then we always mean $\pi = y/(y+z)$.

By use of Assumption (R2), it is easy to prove the following lemma.

Lemma 21.3.7. *There exist some functions* $\underline{\vartheta}, \overline{\vartheta} : (-\lambda^{-1}, \mu^{-1}) \to \mathbb{R}$ *such that*

$$\mathcal{A}(y,z) = \{\theta\,;\, \underline{\vartheta}(\pi) < \theta/(y+z) < \overline{\vartheta}(\pi)\}.$$

The same result holds for $\mathcal{A}_{N-1}$ *replacing* $(-\lambda^{-1}, \mu^{-1})$ *by* $\mathbb{R}$, *i.e.* (λ, μ) *by* $(0,0)$.

Then the interval $(\underline{\vartheta}(\cdot), \overline{\vartheta}(\cdot))$ will be a function of Π_n for $(Y_n, Z_n) \in \mathcal{S}$. Note that $\overline{\vartheta}(\pi)$ may be negative (if π is too large) and $\underline{\vartheta}(\pi)$ may be positive (if π is too small).

We will use the log-utility and consider the following maximization problem:

$$G_n^*(y,z) := \sup E[\log(Y_N + Z_N) \,|\, Y_n = y,\, Z_n = z], \tag{21.11}$$

where the supremum is taken over all admissible trading strategies. The expectation in (21.11) is well-defined. In fact, for given (y,z), the integrand $\log(Y_N + Z_N)$ is bounded from above. For that fact it is sufficient to consider the case without transaction costs which was treated in Korn and Schäl [15, Theorem 4.12]. From dynamic programming we know that we can restrict to Markov policies where $\Delta_n = \delta_n(Y_n, Z_n)$. There a trading strategy will be described by a Markov policy $\{\delta_n, n = 0, \ldots, N-1\}$ if the decision rule δ_n is a function on $\mathcal{S}$ with $\delta_{N-1}(y,z) \in \mathcal{A}_{N-1}(y,z)$ and $\delta_n(y,z) \in \mathcal{A}(y,z)$ for $n < N-1$. Set

$$G_n(y,z) := E[G_{n+1}^*(yR_{n+1}, z)]. \tag{21.12}$$

Then the following optimality equation holds:

$$G_n^*(y,z) = \max_\theta G_n(y+\theta, z - K(\theta)), \tag{21.13}$$

where θ runs through $\mathcal{A}_{N-1}(y,z)$ for $n = N-1$ and through $\mathcal{A}(y,z)$ for $n < N-1$. The optimality criterion states (see e.g. [1, Theorem 2.3.8]): If there are maximizers $\theta^* = \delta_n(y,z)$ such that

$$G_n(y+\theta^*, z - K(\theta^*)) = \max_\theta G_n(y+\theta, z - K(\theta)), \tag{21.14}$$

then $\{\delta_n\}$ defines an optimal Markov policy.

Definition 21.3.8. We call a line $\{(y+\theta, z-(1-\mu)\theta) : \theta \in \mathbb{R}\}$ a *sell-line* and a line $\{(y+\theta, z-(1+\lambda)\theta) : \theta \in \mathbb{R}\}$ a *buy-line*.

We can now state the main theorem on the structure of the optimal Markov policy.

Theorem 21.3.9. *For* $n = N-1, \ldots, 1, 0$ *we have*

a. There exist numbers $-1/\lambda < a_n \le b_n < 1/\mu$ *such that the following holds: There exists an optimal Markov policy* $\{\delta_n\}$ *where* $\{\delta_n\}$ *is defined by*

(i) $\delta_n = 0$ *on the* no-trading cone $\mathcal{T}_n^{\text{notr}} := \{(y,z) \in \mathcal{S} : a_n \le \pi \le b_n\}$,
(ii) $\delta_n(y,z) = \theta < 0$ *on the* sell cone $\mathcal{T}_n^{\text{sell}} := \{(y,z) \in \mathcal{S}; b_n < \pi < 1/\mu\}$ *such that* $(y+\theta, z-(1-\mu)\theta)$ *is situated on the ray* $\{(\alpha b_n, \alpha(1-b_n)) : \alpha \ge 0\}$,
(iii) $\delta_n(y,z) = \theta > 0$ *on the* buy cone $\mathcal{T}_n^{\text{buy}} := \{(y,z) \in \mathcal{S} : -1/\lambda < \pi < a_n\}$ *such that* $(y+\theta, z-(1+\lambda)\theta)$ *is situated on the ray* $\{(\alpha a_n, \alpha(1-a_n)) : \alpha \ge 0\}$.

b. $G_n^*(\alpha y, \alpha z) = \log \alpha + G_n^*(y,z)$ *for* $\alpha > 0$ *and* $G_n^*(y,z)$ *is concave and isotone in each component.*
c. On the sell-line through (y,z), G_n *attains its maximum in a point* $(\alpha b_n, \alpha(1-b_n))$ *for some* $\alpha \in \mathbb{R}$. *On the buy-line through* (y,z), G_n *attains its maximum in a point* $(\alpha a_n, \alpha(1-a_n))$ *for some* $\alpha \in \mathbb{R}$.
d. The sell cone and the buy cone (and of course the no-trading cone) are not empty.

Condition (R3) is only used for part (d) in Theorem 21.3.9, but it will play an important role in Sects. 21.4 and 21.5. Now the theorem has the following interpretation. Selling can be interpreted as walk on a sell-line in the (y,z)-plane. For (y,z) in the sell-cone, optimal selling then means to walk on a sell-line (starting in (y,z)) until one reaches the boundary of the no-trading-cone. The situation for the buy-cone is similar. $\mathcal{T}_n^{\text{notr}} \cup \{0\}$ is a closed convex cone and $\mathcal{T}_n^{\text{notr}}$ degenerates to the Merton-line if $\mu = \lambda = 0$. In the present general case the boundaries of $\mathcal{T}_n^{\text{notr}}$ may be called the *two Merton-lines*. The proof of the theorem is given in Appendix 21.6. A similar result holds for the power utility function $U_\gamma(w) = \gamma^{-1} w^\gamma$, $0 \ne \gamma < 1$ (see Sass and Schäl [23]).

21.4 Martingale Properties of the Optimal Markov Decision Process

Given the optimal policy $\{\delta_n\}$ from Theorem 21.3.9, the initial value (y,z), and the sequence $R_n(\omega)$, $n \ge 1$, we can construct the state process $(Y_n(\omega), Z_n(\omega))$, $n \ge 0$. In the sequel we will only consider this process $\{(Y_n, Z_n), n = 0, \dots, N\}$ determined by the optimal policy. In this section we want to prove a martingale property of the optimal Markov decision process which is important for the financial application. In the model without transaction costs, $\{(Y_n + Z_n)^{-1}\}$ is a martingale. In the presence of transaction costs one has to modify Y_n by a factor ρ_n which is close to one if the transaction costs are small. Our main goal will be to prove that $\{(\rho_n Y_n + Z_n)^{-1}\}$ is a martingale then.

Besides the risky fraction Π_n we will consider the risky fraction after trading $\overline{\Pi}_n$ defined by

$$\overline{\Pi}_n := \overline{Y}_n / (\overline{Y}_n + \overline{Z}_n). \tag{21.15}$$

Further we introduce

$$\hat{\Pi}(\pi, r) := \frac{\pi r}{\pi r + 1 - \pi}. \tag{21.16}$$

Then we obtain from Theorem 21.3.9:

$$\overline{\Pi}_n = 1_{\{\Pi_n \le a_n\}} a_n + 1_{\{a_n < \Pi_n < b_n\}} \Pi_n + 1_{\{\Pi_n \ge b_n\}} b_n \tag{21.17}$$

$$\Pi_{n+1} = \overline{Y}_n R_{n+1} / (\overline{Y}_n R_{n+1} + \overline{Z}_n) = \hat{\Pi}(\overline{\Pi}_n, R_{n+1}). \tag{21.18}$$

By the definition of (Y_n, Z_n) above, we know that (21.11) becomes

$$G_n^*(y,z) = E[\log(Y_N + Z_N) \,|\, Y_n = y, Z_n = z]. \tag{21.19}$$

Then we have $G_{N-1}^*(y,z) = G_{N-1}(y,z)$ for (y,z) in the no-trading cone $a_{N-1} \le \pi \le b_{N-1}$ where

$$G_{N-1}(y,z) = E[\log(yR_N + z)]. \tag{21.20}$$

Definition 21.4.1. We define $H_N := Y_N + Z_N = \rho_N Y_N + Z_N$, where $\rho_N := 1$, and for $n = N-1, \ldots, 0$

$$\rho_n := E[\rho_{n+1} R_{n+1} H_{n+1}^{-1} \,|\, \mathcal{F}_n] / E[H_{n+1}^{-1} \,|\, \mathcal{F}_n],$$
$$H_n := \rho_n Y_n + Z_n.$$

Remark 21.4.2. In Definition 21.4.1, ρ_n is well-defined since H_{n+1} is positive and bounded away from zero given $(\overline{Y}_n, \overline{Z}_n) = (y,z) \in \Gamma_N$ (and Γ, respectively).

Lemma 21.4.3. *One can write $\rho_n = \hat{\rho}_n(\Pi_n)$ for some function $\hat{\rho}_n$, i.e. ρ_n depends on the history only through Π_n.*

a. For $a_n \le \pi \le b_n$

$$\hat{\rho}_n(\pi) = E[\hat{\rho}_{n+1}(\hat{\Pi}(\pi, R_{n+1})) R_{n+1} H_{n+1}^{-1}] / E[H_{n+1}^{-1}],$$

where $H_{n+1} = \hat{\rho}_{n+1}(\hat{\Pi}(\pi, R_{n+1}))\pi R_{n+1} + 1 - \pi$.

b. For $\pi \le a_n$ we have $\hat{\rho}_n(\pi) = \hat{\rho}_n(a_n)$.

c. For $\pi \ge b_n$ we have $\hat{\rho}_n(\pi) = \hat{\rho}_n(b_n)$.

Proof. For $n = N$ we set $\hat{\rho}_N = 1$. For the induction step $n+1 \to n$ let $\overline{\Pi}_n = \pi$ and $\overline{Y}_n + \overline{Z}_n = x$ be fixed. Then $\rho_n = E[\hat{\rho}_{n+1}(\hat{\Pi}(\pi, R_{n+1}) R_{n+1} H_{n+1}^{-1} \,|\, \mathcal{F}_n] / E[H_{n+1}^{-1} \,|\, \mathcal{F}_n]$, where $H_{n+1} = \hat{\rho}_{n+1}(\Pi_{n+1}) Y_{n+1} + Z_{n+1} = x\big(\hat{\rho}_{n+1}(\hat{\Pi}(\pi, R_{n+1}))\pi R_{n+1} + 1 - \pi\big)$. Thus ρ_n is in fact a function of $\overline{\Pi}_n = \pi$ and thus $\hat{\rho}_n$ a function of Π_n.

Now (b) and (c) follow in view of (21.17). □

Lemma 21.4.4. *$\hat{\rho}_n$ is continuous.*

Proof. We know that $\rho_N \equiv 1$ is continuous. We will prove now that $\hat{\rho}_n$ is continuous if $\hat{\rho}_{n+1}$ is continuous. By Lemma 21.4.3(b), (c), $\hat{\rho}_n$ is continuous for $\pi \le a_n$ and for $\pi \ge b_n$. For $a_n \le \pi \le b_n$ the statement follows from Lemma 21.4.3(a), since $\hat{\Pi}(\pi, r)$ is continuous in π. □

Theorem 21.4.5.

a. $\{H_n^{-1}, n = 0,\dots,N\}$ *is a martingale,*
b. $1-\mu \le \rho_n \le 1+\lambda$, $n = 0,\dots,N$.

The proof is given in Appendix 21.6.

21.5 Price Systems and the Numeraire Portfolio

Price Systems and Martingale Measures Q

In this section discount factors play an important role. Then the theory seems to become more transparent if we write the discount factor B_n^{-1} explicitly. We are interested in an alternative probability measure Q with density $q = dQ/dP$ w.r.t P, where Q has the same null sets as P, i.e. Q and P are equivalent. Then we have

$$q > 0 \text{ a.s.} \quad \text{and} \quad E[q] = 1, \quad Q(A) = \int_A q\,dP \quad \text{for } A \in \mathcal{F}. \tag{21.21}$$

Now consider a contingent claim (Y^C, Z^C) maturing in N and split into a contingent claim Y^C for the stock account and a contingent claim Z^C for the bank account. We want to find a price $\mathrm{pr}(Y^C, Z^C)$ for (Y^C, Z^C) and will use the following approach (ansatz) if (Y^C, Z^C) is bounded or if $Y^C + Z^C \ge 0$:

$$\mathrm{pr}(Y^C, Z^C) = E_Q\left[B_N^{-1}(Y^C + Z^C)\right] = E\left[q\,B_N^{-1}(Y^C + Z^C)\right]. \tag{21.22}$$

Theorem 21.5.1. $\mathrm{pr}(\cdot)$ *as given by (21.22) defines a price system, i.e. one has* $\mathrm{pr}(Y^C, Z^C) > 0$ *for any* (Y^C, Z^C) *with the properties*

$$Y^C + Z^C \ge 0 \quad a.s., \qquad P(Y^C + Z^C > 0) > 0. \tag{21.23}$$

The proof of Theorem 21.5.1 is given by Kusuoka [18] for finite probability spaces. There it is shown that the form (21.22) is also necessary for a consistent price system as defined in Theorem 21.5.3 below. See also Sass and Schäl [23]. We will write

$$q_n := E[q \,|\, \mathcal{F}_n]. \tag{21.24}$$

Then $\{q_n\}$ is the *density process* and is a martingale under P by definition. Now we define $\{\rho_n\}$ given $q = q_N$, $\rho_N = 1$. It will turn out that the process will agree with $\{\rho_n\}$ as defined in Sect. 21.4.

Definition 21.5.2. $q_n \rho_n B_n^{-1} S_n := E[q\,B_N^{-1} S_N \,|\, \mathcal{F}_n]$, (i.e. $\rho_n = E_Q[R_{n+1} \cdots R_N \,|\, \mathcal{F}_n]$).

The equation in parentheses follows from Bayes' rule. Then $\{q_n \rho_n B_n^{-1} S_n\}$ is a martingale under P by definition which also means, in view of Bayes' rule, that $\{\rho_n B_n^{-1} S_n\}$ is a martingale under Q. If there are no transaction costs, i.e. $\lambda = \mu = 0$,

we have under condition (21.25) below $\rho_n = 1$, $1 \le n \le N$. Then the discounted stock price process $\{B_n^{-1} S_n\}$ forms a martingale under the probability measure Q with density q and density process $\{q_n\}$. That is the reason for calling Q a martingale measure then.

Now we define the notion of a consistent price system and give a condition in terms of $\{\rho_n\}$.

Theorem 21.5.3. *Assume for* $1 \le n \le N$

$$1 - \mu \le \rho_n \le 1 + \lambda. \tag{21.25}$$

Then the price system $\mathrm{pr}(\cdot)$ *is* consistent, *i.e.*

$$\mathrm{pr}(Y^C, Z^C) = 1 \quad \textit{for} \quad (Y^C, Z^C) = (0, B_N); \tag{21.26}$$

$$(1-\mu) S_0 \le \mathrm{pr}(Y^C, Z^C) \le (1+\lambda) S_0 \quad \textit{for} \quad (Y^C, Z^C) = (S_N, 0); \tag{21.27}$$

$$\mathrm{pr}(Y^C, Z^C) \le 0 \quad \textit{for} \quad (Y^C, Z^C) = (Y_N, Z_N), \tag{21.28}$$

where (Y_N, Z_N) *is the terminal portfolio under an arbitrary admissible policy with start in* $(Y_0, Z_0) = (0,0)$.

Relation (21.26) is natural. If one starts with 1 unit of bond, then one can be sure to have B_N on the bank account at N. Relation (21.27) is also natural. Let us only consider the case $\lambda = \mu = 0$ without transaction costs. If one starts then with 1 unit of stock, then one can be sure to have S_N on the stock account at N. Relation (21.28) excludes a sort of arbitrage opportunity. Starting with nothing one can never reach a portfolio with a positive price. The proof of Theorem 21.5.3 is given by Kusuoka [18] for finite probability spaces. There it is shown that (21.25) is also necessary for a consistent price system.

The Numeraire Portfolio

Now we can explain the main purpose of the paper in terms of this section. We study the following problem. Can we replace the discount factor B_N^{-1} by a more general one, H_N^{-1}, where H_N is the terminal total wealth under some traded portfolio, and then keep to the original (physical) probability measure in place of Q. Thus we want find an admissible policy with start in (Y_0, Z_0) and with total wealth $H_N = Y_N + Z_N$ at N such that $E[q B_N^{-1}(Y^C + Z^C)] = E[H_N^{-1}(Y^C + Z^C)]$. Then we have to define q by

$$B_N^{-1} q = c\,(Y_N + Z_N)^{-1} = c\, H_N^{-1}, \quad c = E[H_N^{-1} B_N]^{-1}, \tag{21.29}$$

where the case $c = 1$ is of particular interest.

From now on, we return to the setting where $B_n \equiv 1$.

Lemma 21.5.4. *The definition of* $\{\rho_n\}$ *in Sect. 21.4 agrees with Definition 21.5.2 and we have* $q_n = c\, H_n^{-1}$.

We will require that $c = 1$ in Corollary 21.1 below.

Proof. Let (Y_N, Z_N) be the portfolio at N under the optimal policy as in Sect. 21.4. Set $H_N := Y_N + Z_N = \rho_N Y_N + Z_N$, $\rho_n := E[\rho_{n+1} R_{n+1} H_{n+1}^{-1} \,|\, \mathcal{F}_n] / E[H_{n+1}^{-1} \,|\, \mathcal{F}_n]$ as in Definition 21.4.1 and define $H_n := \rho_n Y_n + Z_n$, $n < N$. Then we can conclude from Theorem 21.4.5(a) that

$$\{H_n^{-1}\} \quad \text{is a martingale.} \tag{21.30}$$

Upon setting $q = q_N := c H_N^{-1}$ as above, we obtain $q_n = E[c H_N^{-1} \,|\, \mathcal{F}_n] = c H_n^{-1}$ and $\rho_n H_n^{-1} = \rho_n E[H_{n+1}^{-1} \,|\, \mathcal{F}_n] = E[\rho_{n+1} R_{n+1} H_{n+1}^{-1} \,|\, \mathcal{F}_n]$. This yields

$$\begin{aligned} q_n \rho_n S_n &= c H_n^{-1} \rho_n S_n = c\, S_n E[\rho_{n+1} R_{n+1} H_{n+1}^{-1} \,|\, \mathcal{F}_n] \\ &= c E[\rho_{n+1} S_{n+1} H_{n+1}^{-1} \,|\, \mathcal{F}_n] = E[q_{n+1} \rho_{n+1} S_{n+1} \,|\, \mathcal{F}_n]. \end{aligned}$$

Thus $\{q_n \rho_n S_n\}$ is a martingale under P and the definition of ρ_n in Sect. 21.4 agrees with Definition 21.5.2. □

Now we are allowed to apply Theorem 21.4.5(b) and we get condition (21.25). Hence Theorem 21.5.3 applies and we know that $\mathrm{pr}(Y^C, Z^C) = c\,[H_N^{-1}(Y^C + Z^C)]$ is a consistent price system. For c we have $1 = E[q] = c E[H_N^{-1}] = c H_0^{-1}$ by (21.30). Thus

$$c = H_0 = \rho_0 Y_0 + Z_0. \tag{21.31}$$

For models without transaction costs, one usually starts with one unit of money to get the discount factor. If we do the same in the present case, then we start with $(Y_0, Z_0) = (0, 1)$ and thus with $c = H_0 = 1$. Thus we get the following corollary as main result.

Corollary 21.1. *Let $\{(Y_n, Z_n)\}$ be generated by an optimal policy as in Sect. 21.4. If we start with $(Y_0, Z_0) = (0, 1)$ or more generally with $H_0 = \rho_0 Y_0 + Z_0 = 1$, then a consistent price system is given by*

$$\mathrm{pr}(Y^C, Z^C) = E[(Y_N + Z_N)^{-1}(Y^C + Z^C)].$$

Definition 21.5.5. In the situation of Corollary 21.1 we call the dynamic portfolio $\{(Y_n, Z_n)\}$ a *numeraire portfolio*.

21.6 Conclusive Remarks

Extension 21.6.1. A similar result can be derived for power utility $U_\gamma(x) = x^\gamma/\gamma$ with $U_\gamma'(w) = w^{\gamma-1}$ and $U_\gamma^*(w) = U_\gamma'(w)\, w = w^\gamma$ for $0 \neq \gamma < 1$, where $\gamma = 0$ would correspond to the log-utility. When starting again with $(Y_0, Z_0) = (0, 1)$, one obtains a consistent price system (see Sass and Schäl [23]) by

$$\mathrm{pr}^\gamma(Y^C, Z^C) = E[U_\gamma^*(Y_N + Z_N)]^{-1} E[U_\gamma'(Y_N + Z_N)(Y^C + Z^C)], \tag{21.32}$$

where $\{(Y_n, Z_n)\}$ now is the optimal dynamic portfolio for U_γ. Then (R3) is to be replaced by $E[(R_n - \underline{R})^{\gamma-1}] = E[(\overline{R} - R_n)^{\gamma-1}] = \infty$. Now (21.32) formally corresponds

to formula (21.2), but $Y_N + Z_N$ still depends on the transaction costs. On the one hand, the power utility allows to work with a more general relative risk aversion $1-\gamma$ of the investor. On the other hand we have to work with a probability measure $Q_\gamma \neq P$. In fact, we then have

$$Q_\gamma(A) = \int q_\gamma dP, \quad A \in \mathcal{F}, \quad \text{and} \quad q_\gamma = E[U_\gamma^*(Y_N + Z_N)]^{-1} U_\gamma'(Y_N + Z_N)\tilde{B}_N$$

if we decide for $\tilde{B}_N^{-1}$ as discount factor. We can choose $\tilde{B}_N = B_N$ or $\tilde{B}_N = Y_N + Z_N$ or more generally $\tilde{B}_N = Y_N^0 + Z_N^0$, where $\{(Y_n^0, Z_n^0)\}$ is the dynamic portfolio under any admissible policy $\{\delta_n^0\}$.

Algorithm 21.6.2. The pricing of financial derivatives under proportional transaction costs can now be done efficiently as follows. First, by backward induction one can find numerically the boundaries $a_{N-1}, \ldots, a_0$ and $b_{N-1}, \ldots, b_0$ of the no-trade-region which exist according to Theorem 21.3.9(c). Second, having computed these constants, the dynamic portfolio (Y_n, Z_n), $n = 0, \ldots, N$, under the optimal policy can then be computed forwardly for any path of the stock prices. These computations are independent of the specific claims we want to price. For any financial derivative $C = (Y^C, Z^C)$ we find a price according to Corollary 21.1. Since this price system is consistent, the resulting price does not lead to arbitrage. This price is preference based. Since it depends on the log-optimal portfolio it corresponds to an investor with logarithmic utility which has relative risk aversion 1. Different relative risk aversions $1-\gamma > 0$ can be covered by using power utility functions as in Extension 21.6.1. Also for these the computation is efficient in the sense that the optimal policy can be computed first and then prices for any claim can be found by taking expectations as in (21.32).

The formulation of a utility optimization problem in discrete time $0 \leq n \leq N$ for a financial market as a Markov decision model is now classical. This is also true for models with transaction costs (see Kamin [12], Constantinides [5]). However we add some new features. In particular, we use the first order condition of the optimal action as for (21.2). For that argument, it is necessary that the optimal action lies in the interior of the action space which is guaranteed by working with open action spaces. In fact, the first order condition leads to the martingale property in Theorem 21.4.5(a).

In Lemma 21.5.4, $\{H_n^{-1}\}$ is identified as the density process $\{q_n\}$ and we see that the martingale property for $\{H_n^{-1}\}$ must necessarily hold. Moreover this property is also used in Lemma 21.5.4 to show that $\{H_n^{-1}\rho_n S_n\}$ is a martingale as well.

The paper treats a financial model with one stock (and one bond). But models with d stocks ($d > 1$) and transition costs play an important role and one can ask for extensions of the present results to models with several stocks. Numerical results show that for $d > 1$ the structure of the optimal policy may be complicated. Without knowing the structure of the optimal policy, one can however prove by use of the methods of Kallsen and Muhle-Karbe [11] that the main result remains true for models where the underlying probability space is finite. In fact, for such models the optimal policy defines a dynamic portfolio which is a numeraire portfolio. It seems to be unknown whether this extends to infinite probability spaces.

Appendices

Proof of Theorem 21.3.9

We will use backward induction in the dynamic programming procedure. Thus stage $N-1$ will be the stage of the induction start. We set

$$g_N(y,z) := \log(y+z) \quad \text{for} \quad (y,z) \in \mathcal{S}_N,$$
$$G_{N-1}(y,z) := E[g_N(yR_N,z)] \quad \text{for} \quad (y,z) \in \Gamma_N.$$

For the induction, we now consider the following more general optimization problem: The *gain function* $g(y,z)$ is any function on $\mathcal{S}_N$ satisfying the following hypotheses:

$$g \text{ is isotone in each component, concave, and } g(\alpha y,\alpha z) = \log(\alpha) + g(y,z) \text{ for } \alpha > 0. \tag{21.33}$$

Moreover we will use the following technical assumption:

$$\begin{aligned} &\text{For } 0 \neq (y',z') \in \partial\mathcal{S}_N \text{ there is a neighborhood } \mathcal{N} \text{ of } (y',z') \\ &\text{such that } g(y,z) = \log(y+z) + \text{const on } \mathcal{N}. \end{aligned} \tag{21.34}$$

Obviously (21.33) and (21.34) generalize the case where $g = g_N$. Define the *objective function* by $G(y,z) := E[g(yR_N,z)], \quad (y,z) \in \Gamma_N$,

$$\begin{aligned} G^*(y,z) &:= \sup_{\theta \in \mathcal{A}_{N-1}(y,z)} G(y+\theta, z-K(\theta)) \\ &= \sup_{\underline{\vartheta}(\pi) < \vartheta < \overline{\vartheta}(\pi)} G(y+\vartheta(y+z), z-K(\vartheta(y+z))) \end{aligned}$$

for $(y,z) \in \mathcal{S}$. From dynamic programming we know that $\theta^* = \delta^*(y,z)$ is optimal in state (y,z) at stage $N-1$ if $G^*(y,z) = G(y+\theta^*, z-K(\theta^*))$ where G^* is the optimal gain function at stage $N-1$ for the special case "$g = g_N$". G^* will inherit the properties of g.

Lemma 21.7.1.

a. $G(y,z)$ *is concave and isotone in each component and*

$$G(\alpha y,\alpha z) = \log(\alpha) + G(y,z) \text{ for } \alpha > 0.$$

b. (Concavity and Isotony of $\mathcal{A}_{N-1}$*)*

(i) If $\theta_i \in \mathcal{A}_{N-1}(y_i,z_i)$, $\gamma_i > 0$, $i = 1,2$, $\gamma_1 + \gamma_2 = 1$, *then* $\sum \gamma_i\theta_i \in \mathcal{A}_{N-1}(\sum \gamma_i(y_i,z_i))$.

(ii) $\mathcal{A}_{N-1}$ *is increasing in each component, i.e.,* $\mathcal{A}_{N-1}(y_1,z_1) \subseteq \mathcal{A}_{N-1}(y_2,z_2)$ *for* $y_1 \leq y_2$, $z_1 \leq z_2$.

c. $G^*(\alpha y, \alpha z) = \log(\alpha) + G^*(y,z)$ *for* $\alpha > 0$.

The simple proof is omitted. It makes use of the convexity of K and the relation

$$\theta \in \mathcal{A}_{N-1}(\alpha y, \alpha z) \quad \text{if and only if} \quad \theta \in \{\vartheta\, \alpha(y+z) : \underline{\vartheta}(\pi) < \vartheta < \overline{\vartheta}(\pi)\}.$$

The hypothesis (21.33) for G^* in place of g will now follow from the following fact.

Proposition 21.7.2. $G^*(y,z)$ *is concave and isotone in each component.*

The arguments of the proof are standard in dynamic programming (see Bäuerle and Rieder [1]). The proof of Lemma 21.7.1(c) (also standard) would show that $\alpha\theta^*$ is a maximizer for

$$G^*(\alpha y, \alpha z) = \sup_{\theta \in \mathcal{A}_{N-1}(\alpha y, \alpha z)} G(\alpha y + \theta, \alpha z - K(\theta)),$$

if θ^* is a maximizer for $G^*(y,z)$. Therefore we can restrict attention to the case $y+z=1$ and we will consider $(y,z) = (\pi, 1-\pi) \in \mathcal{S}$. Now fix some π, say $\pi = \frac{1}{2}$, and consider the following sell-line ℓ^{sell} and buy-line ℓ^{buy} in the (y,z)-plane parametrized by ϑ:

$$\ell^{\text{sell}} = \left\{ \left(\frac{1}{2} + \vartheta, \frac{1}{2} - (1-\mu)\vartheta \right) : \vartheta \in \mathbb{R} \right\},$$
$$\ell^{\text{buy}} = \left\{ \left(\frac{1}{2} + \vartheta, \frac{1}{2} - (1+\lambda)\vartheta \right) : \vartheta \in \mathbb{R} \right\}.$$

Proposition 21.7.3. *The maxima of* G *on* $\ell^{\text{sell}} \cap \Gamma_N$ *and on* $\ell^{\text{buy}} \cap \Gamma_N$ *are attained.*

Proof. (i) We will only consider ℓ^{sell} and set $R := R_N$. We know that $(y_N, z_N) := (\frac{1}{2} + \overline{\vartheta}, \frac{1}{2} - (1-\mu)\overline{\vartheta}) \in \partial\Gamma_N$ where $\overline{\vartheta} := \overline{\vartheta}(\frac{1}{2})$. Now set $s := \vartheta - \overline{\vartheta} < 0$ and define the concave function

$$I(\vartheta) := G(\frac{1}{2} + \vartheta, \frac{1}{2} - (1-\mu)\vartheta) = I(s + \overline{\vartheta}) = E[g((y_N + s)R, z_N - (1-\mu)s)].$$

We will show below that the one-sided derivative $\frac{d^-}{d\vartheta} I(\vartheta) = \frac{d^-}{ds} I(s + \overline{\vartheta})$ is negative if ϑ is close to $\overline{\vartheta}$. This fact implies that $I(\vartheta)$ is decreasing if ϑ approaches $\overline{\vartheta}$ and thus $I(\vartheta)$ cannot be close to $\sup I$. We only consider the case where $y_N > 0$, $z_N < 0$. A similar argument will hold for the other boundary point of ℓ^{sell}.
(ii) Now we study $\frac{d^-}{ds} I(s + \overline{\vartheta}) = E[\frac{d^-}{ds} g((y_N + s)R, z_N - (1-\mu)s)]$, where the equality follows from the monotone convergence theorem and the concavity. If $0 < \eta < y_N \wedge (1 - \mu - \underline{R})$ is small, then $((y_N + s)r, z_N - (1-\mu)s)$ is close to $(y_N \underline{R}, z_N) \in \partial\mathcal{S}_N$ for $-\eta < s < 0$ and $\underline{R} \le r \le \underline{R} + \eta$. By hypothesis (21.34) we then may assume that

$$g(y,z) = \log(y+z) + \text{const} \quad \text{for} \quad (y,z) = ((y_N + s)r, z_N - (1-\mu)s). \tag{21.35}$$

In order to use Fatou's lemma we will show that $\frac{d^-}{ds} g((y_N+s)R, z_N-(1-\mu)s)$ is bounded from above by some c, say. Indeed we know from (21.33) that

$$g((y_N+s)r, z_N-(1-\mu)s) = \log((y_N+s)r) + g(1, q(s)/r)$$

for $q(s) := (z_N-(1-\mu)s)/(y_N+s)$. Note that $q(s)$ is decreasing. Now $g(1,q(s)/r)$ inherits this property since g is increasing; therefore its one-sided derivative $\frac{d^-}{ds}$ is bounded from above by zero. The derivative $\log((y_N+s)r)$ is obviously bounded from above. Now we can conclude

$$\limsup_{\vartheta\to\overline{\vartheta}} \frac{d^-}{d\vartheta} I(\vartheta) \le E[\limsup_{s\nearrow 0} \frac{d^-}{ds} g((y_N+s)R, z_N-(1-\mu)s)] \le A + cP(R > \underline{R}+\eta),$$

where $A := E\left[1_{\{R\le \underline{R}+\eta\}}(y_N R+z_N)^{-1}(R-(1-\mu))\right]$ in view of (21.35). There we have $R-(1-\mu) \le (\underline{R}+\eta)-(1-\mu) \le \underline{R}-(1-\mu)+\eta < 0$. Now $(y_N, z_N) \in \partial\Gamma_N$ implies $\underline{R} y_N + z_N = 0$ and thus $(y_N R+z_N)^{-1} = (y_N(R-\underline{R}))^{-1}$. From (R3) we then know that $E[1_{\{R\le \underline{R}+\eta\}}(y_N R+z_N)^{-1}] = \infty$. This finally implies $A = -\infty$. □

Definition 21.7.4. Let (y_-, z_-) and (y_+, z_+) be maximum points of G on $\ell^{\text{sell}} \cap \Gamma_N$ and $\ell^{\text{buy}} \cap \Gamma_N$, respectively. If there is more than one, define (y_-, z_-) (resp. (y_+, z_+)) such that the y-value y_- is maximal (resp. y_+ is minimal). Set $a := y_+/(y_+ + z_+)$, $b := y_-/(y_- + z_-)$.

Then in view of Lemma 21.7.1 we have for each $\alpha > 0$

$$\begin{aligned} &G(\alpha y_-, \alpha z_-) \ge G(\alpha y_- + \theta, \alpha z_- - (1-\mu)\theta) \text{ for all } \theta \text{ and } \text{“>” if } \theta > 0 \\ &G(\alpha y_+, \alpha z_+) \ge G(\alpha y_+ + \theta, \alpha z_+ - (1+\lambda)\theta) \text{ for all } \theta \text{ and } \text{“>” if } \theta < 0. \end{aligned} \tag{21.36}$$

Lemma 21.7.5. $a \le b$.

Since the proof is similar to the proofs in the literature (Sass and Schäl [23] applies literally), it will be omitted. We will now study the following non-empty cones.

Definition 21.7.6. $\mathcal{T}^{\text{sell}} := \{(y,z) \in \mathcal{S}; b < \pi < 1/\mu\}$,
$\mathcal{T}^{\text{buy}} := \{(y,z) \in \mathcal{S}; -1/\lambda < \pi < a\}$,
$\mathcal{T}^{\text{notr}} := \{(y,z) \in \mathcal{S}; a \le \pi \le b\} = \mathcal{S} \setminus (\mathcal{T}^{\text{sell}} \cup \mathcal{T}^{\text{buy}})$.

By the definition of $y_\pm$, the interval $[a,b]$ is chosen as large as possible. Thus one does not need to trade under the optimal policy if it is not absolutely necessary.

Proposition 21.7.7. *For $(y,z) \in \mathcal{T}^{\text{notr}}$, it is optimal not to buy and not to sell.*
For $(y,z) \in \mathcal{T}^{\text{sell}}$ it is optimal to sell $|\theta_-|$ where $\theta_- = \delta^*(y,z)$ is defined by (21.37) below.
For $(y,z) \in \mathcal{T}^{\text{buy}}$ it is optimal to buy θ_+ where $\theta_+ = \delta^*(y,z)$ is defined by (21.38) below.

Proof. If $(y,z) \in \mathcal{T}^{\text{sell}}$, then

$$(y+\theta_-, z-(1-\mu)\theta_-) = \alpha'(b,1-b) = \alpha(y_-,z_-) \in \alpha\,\ell^{\text{sell}} \tag{21.37}$$

for some α, $\alpha' > 0$, $\theta_- < 0$. As a consequence

$$\begin{aligned} G(y+\theta_-, z-(1-\mu)\theta_-) &= G(\alpha y_-, \alpha z_-) \\ &= \max_{\theta} G(\alpha y_- + \theta, \alpha z_- - (1-\mu)\theta) \\ &= \max_{\theta'} G(y+\theta', z-(1-\mu)\theta') \\ &\geq \max_{\theta' \geq 0} G(y+\theta', z-(1+\lambda)\theta') \end{aligned}$$

in view of Lemma 21.7.1(c) and (21.36). Since

$$G^*(y,z) = \max\{\sup_{\theta \geq 0} G(y+\theta, z-(1+\lambda)\theta), \sup_{\theta \leq 0} G(y+\theta, z-(1-\mu)\theta)\},$$

we conclude that $G(y+\theta_-, z-(1-\mu)\theta_-) = G^*(y,z)$. Hence it is optimal to sell $|\theta_-|$ (i.e. buy $\theta_- < 0$) in state (y,z).

Now let $(y,z) \notin \mathcal{T}^{\text{sell}}$ Then $(y,z) = (\alpha y_- + \theta_-, \alpha z_- - (1-\mu)\theta_-)$ for some $\alpha > 0$, $\theta_- \leq 0$. Now $G(\alpha y_- + \theta, \alpha z_- - (1-\mu)\theta)$ is concave in θ. Then for $\varepsilon > 0$ we know that $G(\alpha y_-, \alpha z_-) \geq G(y,z) \geq G(y-\varepsilon, z-(1-\mu)(-\varepsilon))$ Therefore "no selling" is as least as good as "selling any amount ε" in state (y,z).

Analogous results hold for $\mathcal{T}^{\text{buy}}$ where we define θ_+ for $(y,z) \in \mathcal{T}^{\text{buy}}$ by

$$(y+\theta_+, z-(1+\lambda)\theta_+) = \alpha'(a,1-a) = \alpha(y_-,z_-) \tag{21.38}$$

for some α, $\alpha' > 0$, $\theta_+ > 0$. □

Corollary 21.2.

a. Let (y,z) be in the closure of $\mathcal{T}^{\text{sell}}$. Then

$$G^*(y,z) = \log((1-\mu)y+z) + G(b,1-b) - \log(1-\mu b).$$

b. Let (y,z) be in the closure of $\mathcal{T}^{\text{buy}}$. Then

$$G^*(y,z) = \log((1+\lambda)y+z) + G(a,1-a) - \log(1+\lambda a)$$

c. For $(y,z) \in \mathcal{T}^{\text{notr}}$ we have $G^*(y,z) = G(y,z)$.

Proof. We only consider (a). By continuity it is sufficient to consider $(y,z) \in \mathcal{T}^{\text{sell}}$. Then it is optimal to sell $|\theta_-|$ yielding according to (21.37)

$$G^*(y,z) = G(y+\theta_-, z-(1-\mu)\theta_-) = G(\alpha b, \alpha(1-b)) = \log(\alpha) + G(b,1-b).$$

From $(y+\theta_-, z-(1-\mu)\theta_-) = (\alpha b, \alpha(1-b))$ we get $\alpha = ((1-\mu)y+z)/(1-\mu b)$. □

From the corollary we conclude that G^* and $\mathcal{S}$ satisfy hypothesis (21.34) in place of g and $\mathcal{S}_N$.

Now we can start the induction step of dynamic programming in order to find an optimal trading strategy $\{\delta_n, 0 \le n < N\}$ which is known to be Markovian, i.e. δ_n is a function of the state $(y,z) \in \mathcal{S}$ in stage n. Upon choosing $g = g_N$, $G = G_{N-1}$ (defined as above), we obtain $\delta_{N-1} := \delta^*$ where δ^* is also defined as above. As G^* satisfies the hypothesis imposed on g, we can now repeat the optimization step, if we replace $\mathcal{S}_N$ by $\mathcal{S}$ and $\mathcal{A}_{N-1}$ by $\mathcal{A}(y,z) := \{\theta : (y+\theta, z-K(\theta)) \in \Gamma\}$.

Proof of Theorem 21.4.5

From now on we use the notion martingale for a martingale under P (and not under Q) and we write $E_n[\cdot] := E[\cdot\,|\,\mathcal{F}_n]$ for the conditional expectations given $R_1, \ldots, R_n$.

Induction Start

Set $R = R_N$, $a = a_{N-1}$, $b = b_{N-1}$, $\hat{G}(y,z) = G_{N-1}(y,z) = E[\log(yR+z)]$.

Lemma 21.7.8. $\frac{\partial}{\partial\theta}\hat{G}(y+\theta, z-k\theta)|_{\theta=0} = E[(R-k)(yR+z)^{-1}]$ *for* $k > 0$.

Proof. We will prove

$$\frac{\partial^{\pm}}{\partial\theta}\hat{G}(y+\theta, z-k\theta)|_{\theta=0} = E[(R-k)(yR+z)^{-1}] \quad \text{for} \quad k > 0. \tag{21.39}$$

We know that $\log((y+\theta)R + z - k\theta)$ and thus $\hat{G}(y+\theta, z-k\theta)$ are concave in θ. In $\lim_{\theta\to 0\pm}\frac{1}{\theta}\left(\hat{G}(y+\theta, z-k\theta) - \hat{G}(y,z)\right)$ we only need to interchange lim and expectation which can be justified by monotone convergence. □

Lemma 21.7.9. *Let* $(Y_{N-1}, Z_{N-1}) = (y,z)$, $a \le \pi \le b$. *Then*

a. $E[R(yR+z)^{-1}] \le (1+\lambda)E[(yR+z)^{-1}]$;
b. $E[R(yR+z)^{-1}] \ge (1-\mu)E[(yR+z)^{-1}]$.

Proof. (a) In (y,z) "not to order" is at least as good as "to buy", hence

$$0 \ge \frac{1}{\theta}\left(\hat{G}(y+\theta, z-(1+\lambda)\theta) - \hat{G}(y,z)\right) \quad \text{for} \quad \theta > 0$$

by the optimality criterion (21.14). Part (b) is similar. □

Lemma 21.7.10 (First Order Condition).

a. $E[R(bR+1-b)^{-1}] = (1-\mu)E[(bR+1-b)^{-1}]$;
b. $E[R(aR+1-a)^{-1}] = (1+\lambda)E[(aR+1-a)^{-1}]$.

Proof. (a) By Theorem 21.3.9, $(b, 1-b)$ is a maximum point on the sell-line through $(b, 1-b)$ and $(a, 1-a)$ is a maximum point on the buy-line through $(a, 1-a)$. Now Lemma 21.7.8 applies. □

Lemma 21.7.11.

a. $1-\mu \le \rho_{N-1} \le 1+\lambda$;
b. $\hat{\rho}_{N-1}(a) = 1+\lambda = \hat{\rho}_{N-1}(\pi)$ *for* $\pi \le a$; $\quad \hat{\rho}_{N-1}(b) = 1-\mu = \hat{\rho}_{N-1}(\pi)$ *for* $\pi \ge b$.

Proof. In view of Lemma 21.4.3(b), (c), we only consider the case $(Y_{N-1}, Z_{N-1}) = (y,z)$, $a \le \pi \le b$. Then we have $H_N = yR+z$

We get $\hat{\rho}_{N-1}(\pi) = E[RH_N^{-1}]/E[H_N^{-1}]$ from Lemma 21.4.3 and thus statement (a) from Lemma 21.7.9. In the same way we obtain (b) from Lemma 21.7.10. □

Theorem 21.7.12.

a. $E_{N-1}[H_N^{-1}] = H_{N-1}^{-1}$ *(martingale property of* H^{-1}*)*;
b. $E_{N-1}[\rho_N R_N H_N^{-1}] = \rho_{N-1} H_{N-1}^{-1}$.

Proof. (a) We have

$$\begin{aligned} 1 &= E_{N-1}[H_N H_N^{-1}] = E_{N-1}[(\rho_N Y_N + Z_N) H_N^{-1}] \\ &= \overline{Y}_{N-1} E_{N-1}[\rho_N R_N H_N^{-1}] + \overline{Z}_{N-1} E_{N-1}[H_N^{-1}] \\ &= \left(\rho_{N-1}\overline{Y}_{N-1} + \overline{Z}_{N-1}\right) E_{N-1}[H_N^{-1}] \\ &= \left(\rho_{N-1}(Y_{N-1}+\Delta_{N-1}) + Z_{N-1} - K(\Delta_{N-1})\right) E_{N-1}[H_N^{-1}] \\ &= \left(H_{N-1} + \rho_{N-1}\Delta_{N-1} - K(\Delta_{N-1})\right) E_{N-1}[H_N^{-1}]. \end{aligned}$$

From Lemma 21.7.11(b) we get $\rho_{N-1}\Delta_{N-1} = K(\Delta_{N-1})$ which yields (a).

Part (b) follows now from the definition of ρ_{N-1}. □

Corollary 21.3 (Induction Start). *For* $k > 0$

$$\frac{\partial}{\partial\theta} E_{N-1}[G_N^*((y+\theta)R_N, z-k\theta)]|_{\theta=0} = (\rho_{N-1}-k) H_{N-1}^{-1}$$

where $G_N^*(y,z) = \log(y+z)$.

Proof. Lemma 21.7.8 applies directly, where $H_N = yR_N + z$. □

We thus know that the following induction hypothesis holds for $n = N-1$:

Induction Hypothesis 21.7.13.

i. For $Y_n = y$, $Z_n = z$, $\Pi_n = \pi$

$$\frac{\partial}{\partial\theta} E[G_{n+1}^*((y+\theta)R_{n+1}, z-k\theta)]|_{\theta=0} = (\hat{\rho}_n(\pi)-k) H_n^{-1} \text{ for } a_n \le \pi < b_n;$$

ii. $\hat{\rho}_n(a_n) = 1+\lambda = \hat{\rho}_n(\pi)$ for $\pi \le a_n$; $\quad \hat{\rho}_n(b_n) = 1-\mu = \hat{\rho}_n(\pi)$ for $\pi \le b_n$.

Induction Step "$N > n \to n-1$"

We assume throughout this section that the induction hypothesis holds for $n < N$. Suppose that $Y_{n-1} = y$, $Z_{n-1} = z$ are given. We know that $\Pi_n = \hat{\Pi}(\pi, R_n)$ where $\hat{\Pi}$ is defined by (21.16) and set $G(y,z) := E[G_n^*(yR_n, z)]$, hence $G_{n-1}^*(y,z) = \sup_\theta G(y+\theta, z-K(\theta))$, $\rho_n := \hat{\rho}_n(\pi_n)$. Then we have $H_n = \rho_n y R_n + z$ for $a_{n-1} \le \pi \le b_{n-1}$.

Proposition 21.7.14. *Suppose $a_{n-1} \le \pi \le b_{n-1}$ and $k > 0$. Then*

$$\frac{d}{d\theta} G(y+\theta, z-k\theta)|_{\theta=0} = E_{n-1}[(\rho_n R_n - k)H_n^{-1}] = (\hat{\rho}_{n-1}(\pi) - k)E_{n-1}[H_n^{-1}]$$

Proof. Let y,z be arbitrary. We consider one-sided derivatives. Since $\theta \mapsto G_n^*((y+\theta)R_n, z-k\theta)$ is concave by Theorem 21.3.9, we can interchange lim (i.e. $\frac{d^\pm}{d\theta}$) and $E[\cdot]$ by the monotone convergence theorem. Consider first $\lim_{\theta\to 0+}$.

Then we have to study for fixed $R_n = s$ and hence for fixed $\Pi_n = ys/(ys+z)$

$$\lim_{\theta\to 0+} \frac{1}{\theta} \left(G_n^*((y+\theta)s, z-k\theta) - G_n^*(ys,z)\right). \tag{21.40}$$

Case (i, ii): $\pi_n \ge b_n$ or $\pi_n < a_n$, respectively. We know (by Theorem 21.3.9) that $G_n^*(ys,z) = \log(\ell ys + z) + \text{const}$ with $\ell = 1-\mu$ or $\ell = 1+\lambda$, respectively. By continuity this is also true for $\pi_n = b_n$ and $\pi_n = a_n$. We can write for the limit in (21.40)

$$\begin{aligned}\frac{d^+}{d\theta} \log(\ell(y+\theta)s + z - k\theta)|_{\theta=0} &= (\ell s - k)(\ell ys + z)^{-1} \\ &= (\hat{\rho}_n(\Pi_n)s - k)(\hat{\rho}_n(\Pi_n)ys + z)^{-1} = (\hat{\rho}_n(\Pi_n)s - k)H_n^{-1}.\end{aligned}$$

Case (iii) $a_n \le \pi_n < b_n$. Then $G_n^*(ys,z) = E_n[G_{n+1}^*(ysR_{n+1}, z)]$ by the optimality properties (21.13), (21.14) and Theorem 21.3.9. Hence for small θ

$$\begin{aligned}&\frac{1}{\theta}\left(G_n^*((y+\theta)R_n, z-k\theta) - G_n^*(yR_n, z)\right) \\ &\quad = E\left[\frac{1}{\theta}\left(G_{n+1}^*((y+\theta)sR_{n+1}, z-k\theta) - G_{n+1}^*(ysR_{n+1}, z)\right)\right] \\ &\quad = sE\left[\frac{1}{s\theta}\left(G_{n+1}^*((ys+\theta s)R_{n+1}, z - \frac{k}{s}s\theta) - G_{n+1}^*(ysR_{n+1}, z)\right)\right].\end{aligned}$$

The latter term converges for $\theta\to 0+$ by Induction Hypothesis 21.7.13 (i) to $s(\hat{\rho}_n(\pi_n) - k/s)H_n^{-1} = (s\hat{\rho}_n(\pi_n) - k)H_n^{-1}$.

Altogether for all cases:

$$\lim_{\theta\to 0+} \frac{1}{\theta}\left(G_n^*(ys + s\theta, z - k\theta) - G_n^*(ys,z)\right) = (\hat{\rho}_n(\Pi_n)s - k)H_n^{-1}.$$

Thus we finally obtain

$$\lim_{\theta\to 0+} \frac{1}{\theta}\left(G(y+\theta, z-k\theta) - G(y,z)\right) = E_{n-1}[(\hat{\rho}_n(\pi_n)\cdot R_n - k)H_n^{-1}].$$

The case $\lim_{\theta\to 0-}$ is similar. □

Lemma 21.7.15. $a_{n-1} < b_{n-1}$ *for* $(\lambda,\mu) \neq (0,0)$.

Proof. We will write $a = a_{n-1}$, $b = b_{n-1}$. We must prove that $a \neq b$ since we know $a \leq b$. Assume that $a = b$. Then a and b are maximum points on the buy-line and the sell-line through $(a, 1-a) = (b, 1-b)$, respectively. From Proposition 21.7.14 we then obtain for $y = a = b$, $k \in \{1+\lambda, 1-\mu\}$

$$\frac{d}{d\theta} G(y+\theta, z-k\theta)|_{\theta=0} = E_{n-1}[(\rho_n R_n - k)H_n^{-1}] = 0,$$

hence $E_{n-1}[\rho_n R_n H_n^{-1}] = kE_{n-1}[H_n^{-1}]$. This equation cannot hold for two different values of $k \in \{1+\lambda, 1-\mu\}$. Thus $a < b$. □

Proposition 21.7.16.

a. $1-\mu \leq \rho_{n-1} \leq 1+\lambda$;
b. $\hat{\rho}_{n-1}(a_{n-1}) = 1+\lambda = \hat{\rho}_{n-1}(\pi)$ *for* $\pi \leq a_n$, $\quad \hat{\rho}_{n-1}(b_{n-1}) = 1-\mu = \hat{\rho}_{n-1}(\pi)$ *for* $\pi \geq b_n$.

Proof. By use of Proposition 21.7.14, the proof is similar to that of Lemmata 21.7.11. □

Proposition 21.7.17. *The martingale property of* $\{H_{n-1}^{-1}, H_n^{-1}\}$ *holds:* $E_{n-1}[H_n^{-1}] = H_{n-1}^{-1}$.

Proof. By use of Propositions 21.7.14 and 21.7.16, the proof is the same as the proof of Theorem 21.7.12(a). □

In view of Propositions 21.7.16 and 21.7.17 we thus proved Theorem 21.4.5 for $n-1$ and the proof by induction is finished.

Notation

Since we have a non-stationary model and since we need some concepts (and their notation) from finance, our notation is not always standard and we shall in this appendix relate some of our notation to the concepts of classical MDP.

$\mathcal{S}$ and $\mathcal{S}_N$	state space at time $n < N$ and at time N, respectively,
$(y,z) \in \mathbb{R}^2$	state vector,
$\log(y+z)$	final reward at time N depending on the final state $(Y_N, Z_N) = (y,z)$; the reward at time $n < N$ is 0,
θ	action,
$E[\log(y+\theta)R_N + z - K(\theta))]$	expected one-step reward at time $N-1$ in state (y,z) under action θ,
$\mathcal{A}(y,z), \mathcal{A}_{N-1}(y,z)$	set of actions available in state (y,z) at time $n < N$ and at time N, respectively,
δ_n	decision rule at time n,
$\delta_n(x,y)$	action at time n under decision rule δ_n if in state (y,z),
$\{\delta_0, \ldots, \delta_{N-1}\} = \{\delta_n\}$	policy with decision rule δ_n at time $n = 0, 1, \ldots, N-1$.

Further,

$$P(B' \times B'' \,|\, n, y_{n-1}, z_{n-1}, \theta_{n-1}) = \int_{B' \times B''} P(dy_n, dz_n \,|\, n, y_{n-1}, z_{n-1}, \theta_{n-1})$$
$$= \text{Prob}\left(((y_{n-1} + \theta_{n-1})R_n, z_{n-1} - K(\theta_{n-1})) \in B' \times B''\right)$$

for measurable $B' \times B'' \subseteq \mathbb{R}^2$ is the (non-stationary) transition probability, and

$$E[\log(Y_N + Z_N \,|\, Y_n = y, Z_n = z]$$

is the value function at time n in state (y,z) over $N-n$ future steps under a Markov policy with decision rules $\{\delta_0, \ldots, \delta_{N-1}\}$, where (Y_m, Z_m) for $n < m \leq N$ is described by the random variables $R_{n+1}, \ldots, R_N$ according to

$$Y_{m+1} = (Y_m + \delta_m(Y_m, Z_m))R_m \quad \text{and} \quad Z_{m+1} = Z_m - K(\delta_m(Y_m, Z_m)).$$

Finally, the optimal value function at time n in state (y,z) over $N-n$ future steps is

$$G_n^*(y,z) = \sup E[\log(Y_N + Z_N) \,|\, Y_n = y, Z_n = z],$$

where the supremum is taken over all admissible Markov policies.

References

1. N. Bäuerle, U. Rieder, *Markov Decision Processes with Applications in Finance* (Springer, Berlin, 2011)
2. D. Becherer, The numeraire portfolio for unbounded semimartingales. Finance Stochast. **5**, 327–341 (2001)
3. H. Bühlmann, E. Platen, A discrete time benchmark approach for insurance and finance. ASTIN Bull. **33**, 153–172 (2003)
4. M.M. Christensen, K. Larsen, No arbitrage and the growth optimal portfolio. Stoch. Anal. Appl. **25**, 255–280 (2007)
5. G.M. Constantinides, Multiperiod consumption and investment behaviour with convex transaction costs. Manag. Sci. **25**, 1127–1137 (1979)
6. J. Cvitanić, I. Karatzas, Hedging and portfolio optimization under transaction costs: a martingale approach. Math. Financ. **6**, 133–166 (1996)
7. M.H.A. Davis, Option pricing in incomplete markets, in *Mathematics of Derivative Securities*, ed. By M. Dempster, S. Pliska (Cambridge University Press, Cambridge, 1997), pp. 216–226
8. M.H.A. Davis, A.R. Norman, Portfolio selection with transaction costs. Math. Oper. Res. **15**, 676–713 (1990)
9. T. Goll, J. Kallsen, A complete explicit solution to the log-optimal portfolio problem. Adv. Appl. Probab. **13**, 774–779 (2003)

10. E. Jouini, H. Kallal, Martingales and arbitrage in securities markets with transaction const. J. Econ. Theory **66**, 178–197 (1995)
11. J. Kallsen, J. Muhle-Karbe, On the existence of shadow prices in finite discrete time. Math. Meth. Oper. Res. **73**, 251–262 (2011)
12. J.H. Kamin, Optimal portfolio revision with a proportional transaction costs. Manag. Sci. **21**, 1263–1271 (1975)
13. I. Karatzas, C. Kardaras, The numéraire portfolio in semimartingale financial models. Finance Stochast. **11**, 447–493 (2007)
14. P.F. Koehl, H. Pham, N. Touzi, On super-replication in discrete time under transaction costs. Theory Probab. Appl. **45**, 667–673 (2001)
15. R. Korn, M. Schäl, On value preserving and growth optimal portfolios. Math. Meth. Oper. Res. **50**, 189–218 (1999)
16. R. Korn, M. Schäl, The numeraire portfolio in discrete time: existence, related concepts and applications. Radon Ser. Comput. Appl. Math. **8**, 1–25 (2009). De Gruyter
17. R. Korn, F. Oertel, M. Schäl, The numeraire portfolio in financial markets modeled by a multi-dimensional jump diffusion process. Decisions Econ. Finan. **26**, 153–166 (2003)
18. S. Kusuoka, Limit theorem on option replication with transaction costs. Ann. Appl. Probab. **5**, 198–121 (1995)
19. J. Long, The numeraire portfolio. J. Financ. **44**, 205–209 (1990)
20. M.J.P. Magill, M. Constantinides, Portfolio selection with transaction costs. J. Econ. Theory **13**, 245–263 (1976)
21. E. Platen, A benchmark approach to finance. Math. Financ. **16**, 131–151 (2006)
22. J. Sass, Portfolio optimization under transaction costs in the CRR model. Math. Meth. Oper. Res. **61**, 239–259 (2005)
23. J. Sass, M. Schäl, Numerairs portfolios and utility-based price systems under proportional transaction costs. Decisions Econ. Finan. **37**, 195–234 (2014)
24. W. Schachermayer, The fundamental theorem of asset pricing under proportional transaction costs in finite discrete time. Math. Financ. **14**, 19–48 (2004)
25. M. Schäl, Portfolio optimization and martingale measures. Math. Financ. **10**, 289–304 (2000)
26. M. Schäl, Price systems constructed by optimal dynamic portfolios. Math. Meth. Oper. Res. **51**, 375–397 (2000)
27. S.E. Shreve, H.M. Soner, Optimal investment and consumption with transaction costs. Ann. Appl. Probab. **4**, 609–692 (1994)

Appendix A: Basic Notation for MDP

S	State space
s/s	A state vector/scalar, given a continuous/discrete state space
i	A state, given a countable state space
r	Reward
$r^a(\mathbf{s})$	Expected one step reward/reward rate in state **s**, under action a
c	Costs
$c^a(\mathbf{s})$	Expected one step cost/cost rate in state **s**, under action a
a	Action
$A(\mathbf{s})$	Set of actions available in state **s**, given a continuous/discrete state space
$A(i)$	Set of actions available in state i, given a countable state space
α	Discount factor
δ	Decision rule
δ_t	Decision rule at time t
$\delta_t(\mathbf{s})$	Action at time t, when in state **s**, given a continuous/discrete state space
$\pi = (\delta, \delta, \ldots)$	Stationary Policy
$\pi = (\delta_0, \delta_1, \delta_2, \ldots)$	Policy with decision rule at time $t = 0, 1, 2, \ldots$
P^π	One step transition probability distribution/matrix under policy π
$P(dy\|(\mathbf{s}, a))$	Transition probability/distribution under action $\pi(\mathbf{s}) = a$, in state **s**
$P^\pi(dy\|\mathbf{s})$	Transition probability distribution/matrix under policy π
$p(j\|i, a)$	Transition probability into state j, when in state i, under action a
Q^π	Transition rate (infinitesimal generator) matrix under policy π
$Q^a_{i,j}$	Transition rate from state i into j (countable) under action a
$q(\mathbf{s'}\|(s, a))$	Transition rate from a state s into a state **s'** under action a, given a continuous/discrete state space
V^π_t	Value function under policy π of expected cumulative reward/costs over t steps (up to time t)
$V^\pi_t(\mathbf{s})$	Value function under policy π of expected cumulative reward /costs over t steps (up to time t) starting in state **s** at time 0
$V^*_t(\mathbf{s})$ or $V_t(\mathbf{s})$	Optimal value function of expected cumulative reward/costs over t steps up to time t, starting in state **s** at time 0
V^π_α	Discounted value function under policy π
V^*_α or V_α	Optimal discounted value function
$G^\pi(\mathbf{s}), g^\pi$ (if ergodic)	Average expected reward/cost function/value under policy π
$G^*(\mathbf{s})$ or $G(\mathbf{s})$, g^* or g (if ergodic)	Optimal average expected reward/cost function/value
$W^\pi(\mathbf{s})/W^*(\mathbf{s})$ or $W(\mathbf{s})$	Expected total reward/costs, $\lim_{t\to\infty} V^\pi_t(\mathbf{s})$, given that the limit exists, under policy π/ an optimal policy
$H^\pi(\mathbf{s})/H^*(\mathbf{s})$ or $H(\mathbf{s})$	Bias of the policy π/ an optimal policy

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4

Appendix B: Dichotomy and Criteria

The table below gives a compact overview of the dichotomy on Discrete or continuous (time and state) modeling aspects. Here the distinction is made based upon the natural or primary description, i.e. not on the used solution procedure e.g. as by uniformization. It also states the performance measure of interest and the optimization criterion used.

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4

Ch.	Topic	Measure	Time	State	Criteria
		R: Rewards C: Costs O: Other	DT: Discrete CT: Continuous	DS: Discrete CS: Continuous	Time Horizon: ITH: Infinite FTH: Finite Costs: AC: Average DC: Discounted
General theory					
1	One-step improvements	R/C O: Delay/Payoff	DT	DS	ITH AC
2	Value function approximation in queueing	O: Delay/Loss	DT	DS	ITH AC
3	ADP: approximate dynamic programming	R: Revenues C: Routing	DT	DS	FTH (ITH: DC)
4	Infinite state queueing	C O: Delay	CT	DS	ITH AC
5	Infinite state structural properties	C O: Delay	DT/CT	DS	ITH AC/DC
Healthcare					
6	Screening and treatment of diseases	O: QALY (see chapter)	DT	DS + CS	FTH - ITH
7	Breast cancer	O: QALY (see chapter)	DT	DS	FTH - ITH
8	Patient appointment scheduling	O: Service level/ Overtime	DT	DS	ITH DC
9	Ambulance dispatching	O: Late arrivals/ Response time	CT	DS	ITH AC
10	Blood supply	O: Outdating	DT	DS	FTH (+ ITH: AC)
Transportation					
11	Airports: noise load management	O: Noise Load	DT	CS	FTH
12	Car park	O: Imbalance	CT	DS	FTH
13	Traffic lights	O: Delays/ Queues	CT	DS	ITH AC
14	Electric vehicles	C: Charging	DT	DS	FTH
Production					
15	Lot scheduling	R: Order Acceptances	DT	DS	ITH AC
16	Fisheries	R: Welfare/profit	DT	CS	ITH DC
17	Flow controllable service rates	O: Delays/workload	CT	DS	ITH AC
Communications					
18	Wireless channel selection	O: Throughput	DT	DS + CS	ITH AC
19	Call center staffing	C: Staffing/ Service level	CT	CS	FTH
20	Query wireless sensoring	O: Freshness/ Response times	CT	DS + CS	ITH AC
Financial modelling					
21	Financial derivatives	R/C: Utility and costs	DT	CS	FTH DC