Dynamic Programming and Optimal Control

Volume I

FOURTH EDITION

Dimitri P. Bertsekas

Massachusetts Institute of Technology

WWW site for book information and orders

http://www.athenasc.com

Athena Scientific, Belmont, Massachusetts

Athena Scientific
Post Office Box 805
Nashua, NH 03060
U.S.A.

Email: info@athenasc.com
WWW: http://www.athenasc.com

Cover photography: Dimitri Bertsekas

Publisher's Cataloging-in-Publication Data

Bertsekas, Dimitri P.
Dynamic Programming and Optimal Control
Includes Bibliography and Index
1. Mathematical Optimization. 2. Dynamic Programming. I. Title.
QA402.5 .B465 2019 519.703 00-91281

ISBN-10: 1-886529-43-4, ISBN-13: 978-1-886529-43-4

Updated 2nd Printing (2020)

ABOUT THE AUTHOR

Dimitri Bertsekas studied Mechanical and Electrical Engineering at the National Technical University of Athens, Greece, and obtained his Ph.D. in system science from the Massachusetts Institute of Technology. He has held faculty positions with the Engineering-Economic Systems Department, Stanford University, and the Electrical Engineering Department of the University of Illinois, Urbana. Since 1979 he has been teaching at the Electrical Engineering and Computer Science Department of the Massachusetts Institute of Technology (M.I.T.), where he is McAfee Professor of Engineering. In 2019, he joined the School of Computing, Informatics, and Decision Systems Engineering at the Arizona State University, Tempe, AZ, as Fulton Professor of Computational Decision Making.

Professor Bertsekas' teaching and research have spanned several fields, including deterministic optimization, dynamic programming and stochastic control, large-scale and distributed computation, artificial intelligence, and data communication networks. He has authored or coauthored numerous research papers and eighteen books, several of which are currently used as textbooks in MIT classes, including "Dynamic Programming and Optimal Control," "Data Networks," "Introduction to Probability," and "Nonlinear Programming."

Professor Bertsekas was awarded the INFORMS 1997 Prize for Research Excellence in the Interface Between Operations Research and Computer Science for his book "Neuro-Dynamic Programming" (co-authored with John Tsitsiklis), the 2001 AACC John R. Ragazzini Education Award, the 2009 INFORMS Expository Writing Award, the 2014 AACC Richard Bellman Heritage Award, the 2014 INFORMS Khachiyan Prize for Life-Time Accomplishments in Optimization, and the 2015 MOS/SIAM George B. Dantzig Prize. In 2018 he shared with his coauthor, John Tsitsiklis, the 2018 INFORMS John von Neumann Theory Prize for the contributions of the research monographs "Parallel and Distributed Computation" and "Neuro-Dynamic Programming." Professor Bertsekas was elected in 2001 to the United States National Academy of Engineering for "pioneering contributions to fundamental research, practice and education of optimization/control theory, and especially its application to data communication networks."

ATHENA SCIENTIFIC

OPTIMIZATION AND COMPUTATION SERIES

1. Rollout, Policy Iteration, and Distributed Reinforcement Learning, by Dimitri P. Bertsekas, 2020, ISBN 978-1-886529-07-6, 376 pages
2. Reinforcement Learning and Optimal Control, by Dimitri P. Bertsekas, 2019, ISBN 978-1-886529-39-7, 388 pages
3. Abstract Dynamic Programming, 2nd Edition, by Dimitri P. Bertsekas, 2018, ISBN 978-1-886529-46-5, 360 pages
4. Dynamic Programming and Optimal Control, Two-Volume Set, by Dimitri P. Bertsekas, 2017, ISBN 1-886529-08-6, 1270 pages
5. Nonlinear Programming, 3rd Edition, by Dimitri P. Bertsekas, 2016, ISBN 1-886529-05-1, 880 pages
6. Convex Optimization Algorithms, by Dimitri P. Bertsekas, 2015, ISBN 978-1-886529-28-1, 576 pages
7. Convex Optimization Theory, by Dimitri P. Bertsekas, 2009, ISBN 978-1-886529-31-1, 256 pages
8. Introduction to Probability, 2nd Edition, by Dimitri P. Bertsekas and John N. Tsitsiklis, 2008, ISBN 978-1-886529-23-6, 544 pages
9. Convex Analysis and Optimization, by Dimitri P. Bertsekas, Angelia Nedić, and Asuman E. Ozdaglar, 2003, ISBN 1-886529-45-0, 560 pages
10. Network Optimization: Continuous and Discrete Models, by Dimitri P. Bertsekas, 1998, ISBN 1-886529-02-7, 608 pages
11. Network Flows and Monotropic Optimization, by R. Tyrrell Rockafellar, 1998, ISBN 1-886529-06-X, 634 pages
12. Introduction to Linear Optimization, by Dimitris Bertsimas and John N. Tsitsiklis, 1997, ISBN 1-886529-19-1, 608 pages
13. Parallel and Distributed Computation: Numerical Methods, by Dimitri P. Bertsekas and John N. Tsitsiklis, 1997, ISBN 1-886529-01-9, 718 pages
14. Neuro-Dynamic Programming, by Dimitri P. Bertsekas and John N. Tsitsiklis, 1996, ISBN 1-886529-10-8, 512 pages
15. Constrained Optimization and Lagrange Multiplier Methods, by Dimitri P. Bertsekas, 1996, ISBN 1-886529-04-3, 410 pages
16. Stochastic Optimal Control: The Discrete-Time Case, by Dimitri P. Bertsekas and Steven E. Shreve, 1996, ISBN 1-886529-03-5, 330 pages

Contents

7. Deterministic Continuous-Time Optimal Control

Appendix A: Mathematical Review

Appendix B: On Optimization Theory

Appendix C: On Probability Theory

Appendix D: On Finite-State Markov Chains

Appendix E: Least Squares Estimation and Kalman Filtering

Appendix F: Formulating Problems of Decision Under Uncertainty

CONTENTS OF VOLUME II

3. Stochastic Shortest Path Problems

4. Undiscounted Problems

5. Average Cost per Stage Problems

Preface

This two-volume book is based on a first-year graduate course on dynamic programming and optimal control that I have taught for over twenty years at Stanford University, the University of Illinois, and the Massachusetts Institute of Technology. The course has been typically attended by students from engineering, operations research, economics, and applied mathematics. Accordingly, a principal objective of the book has been to provide a unified treatment of the subject, suitable for a broad audience. In particular, problems with a continuous character, such as stochastic control problems, popular in modern control theory, are simultaneously treated with problems with a discrete character, such as Markovian decision problems, popular in operations research. Furthermore, many applications and examples, drawn from a broad variety of fields, are discussed.

The book may be viewed as a greatly expanded and pedagogically improved version of my 1987 book "Dynamic Programming: Deterministic and Stochastic Models," published by Prentice-Hall. I have included much new material on deterministic and stochastic shortest path problems, as well as a new chapter on continuous-time optimal control problems and the Pontryagin Minimum Principle, developed from a dynamic programming viewpoint. I have also added a fairly extensive exposition of simulation-based approximation techniques for dynamic programming. These techniques, which are often referred to as "neuro-dynamic programming" or "reinforcement learning," represent a breakthrough in the practical application of dynamic programming to complex problems that involve the dual curse of large dimension and lack of an accurate mathematical model. Other material was also augmented, substantially modified, and updated.

With the new material, however, the book grew so much in size that it became necessary to divide it into two volumes: one on finite horizon, and the other on infinite horizon problems. This division was not only natural in terms of size, but also in terms of style and orientation. The first volume is more oriented towards modeling, and the second is more oriented towards mathematical analysis and computation. I have included in the first volume a final chapter that provides an introductory treatment of infinite horizon problems. The purpose is to make the first volume self-contained for instructors who wish to cover a modest amount of infinite horizon material in a course that is primarily oriented towards modeling, conceptualization, and finite horizon problems,

Many topics in the book are relatively independent of the others. For example Chapter 2 of Vol. I on shortest path problems can be skipped without loss of continuity, and the same is true for Chapter 3 of Vol. I, which deals with continuous-time optimal control. As a result, the book can be used to teach several different types of courses.

(a) A two-semester course that covers both volumes.

(b) A one-semester course primarily focused on finite horizon problems that covers most of the first volume.

(c) A one-semester course focused on stochastic optimal control that covers Chapters 1, 4, 5, and 6 of Vol. I, and Chapters 1, 2, and 4 of Vol. II.

(d) A one-semester course that covers Chapter 1, about 50% of Chapters 2 through 6 of Vol. I, and about 70% of Chapters 1, 2, and 4 of Vol. II. This is the course I usually teach at MIT.

(e) A one-quarter engineering course that covers the first three chapters and parts of Chapters 4 through 6 of Vol. I.

(f) A one-quarter mathematically oriented course focused on infinite horizon problems that covers Vol. II.

The mathematical prerequisite for the text is knowledge of advanced calculus, introductory probability theory, and matrix-vector algebra. A summary of this material is provided in the appendixes. Naturally, prior exposure to dynamic system theory, control, optimization, or operations research will be helpful to the reader, but based on my experience, the material given here is reasonably self-contained.

The book contains a large number of exercises, and the serious reader will benefit greatly by going through them. Solutions to all exercises are compiled in a manual that is available to instructors from the author. Many thanks are due to the several people who spent long hours contributing to this manual, particularly Steven Shreve, Eric Loiederman, Lakis Polymenakos, and Cynara Wu.

Dynamic programming is a conceptually simple technique that can be adequately explained using elementary analysis. Yet a mathematically rigorous treatment of general dynamic programming requires the complicated machinery of measure-theoretic probability. My choice has been to bypass the complicated mathematics by developing the subject in generality, while claiming rigor only when the underlying probability spaces are countable. A mathematically rigorous treatment of the subject is carried out in my monograph "Stochastic Optimal Control: The Discrete Time

Case," Academic Press, 1978,† coauthored by Steven Shreve. This monograph complements the present text and provides a solid foundation for the subjects developed somewhat informally here.

Finally, I am thankful to a number of individuals and institutions for their contributions to the book. My understanding of the subject was sharpened while I worked with Steven Shreve on our 1978 monograph. My interaction and collaboration with John Tsitsiklis on stochastic shortest paths and approximate dynamic programming have been most valuable. Michael Caramanis, Emmanuel Fernandez-Gaucherand, Pierre Humblet, Lennart Ljung, and John Tsitsiklis taught from versions of the book, and contributed several substantive comments and homework problems. A number of colleagues offered valuable insights and information, particularly David Castanon, Eugene Feinberg, and Krishna Pattipati. NSF provided research support. Prentice-Hall graciously allowed the use of material from my 1987 book. Teaching and interacting with the students at MIT have kept up my interest and excitement for the subject.

Dimitri P. Bertsekas
Spring, 1995

† This monograph was republished by Athena Scientific in 1996, and can also be freely downloaded from the author's www site.

Preface to the Second Edition

This second edition has expanded by nearly 30% the coverage of the original. Most of the new material is concentrated in four areas:

(a) In Chapter 4, a section was added on estimation and control of systems with a non-probabilistic (set membership) description of uncertainty. This subject, a personal favorite of the author since it was the subject of his 1971 Ph.D. thesis, has become popular, as minimax and H_∞ control methods have gained increased prominence.

(b) Chapter 6 was doubled in size, to reflect the popularity of suboptimal control and neuro-dynamic programming methods. In particular, the coverage of certainty equivalent, and limited lookahead methods has been substantially expanded. Furthermore, a new section was added on neuro-dynamic programming and rollout algorithms, and their applications in combinatorial optimization and stochastic optimal control.

(c) In Chapter 7, an introduction to continuous-time, semi-Markov decision problems was added in a separate last section.

(d) A new appendix was included, which deals with various formulations of problems of decision under uncertainty. The foundations of the minimax and expected utility approaches are framed within a broader context, and some of the aspects of utility theory are discussed.

There are also miscellaneous additions and improvements scattered throughout the text, and a more detailed coverage of deterministic problems is given in Chapter 1. Finally, a new internet-based feature was added to the book, which extends its scope and coverage. Many of the theoretical exercises have been solved in detail and their solutions have been posted in the book's www page

http://www.athenasc.com/dpbook.html

These exercises have been marked with the symbol (www)

I would like to express my thanks to the many colleagues who contributed suggestions for improvement of the second edition.

Dimitri P. Bertsekas
Fall, 2000

Preface to the Third Edition

The third edition contains a substantial amount of new material, particularly on approximate dynamic programming, which has now become one of the principal focal points of the book. In particular:

(a) The subject of minimax control was developed in greater detail, including a new section in Chapter 1, which connects with new material in Chapter 6.

(b) The section on auction algorithms for shortest paths in Chapter 2 was eliminated. These methods are not currently used in dynamic programming, and a detailed discussion has been provided in a chapter from the author's Network Optimization book. This chapter can be freely downloaded from

http://web.mit.edu/dimitrib/www/net.html

(c) A section was added in Chapter 2 on dynamic programming and shortest path algorithms for constrained and multiobjective problems.

(d) The material on sufficient statistics and partially observable Markov decision problems in Section 5.4 was restructured and expanded.

(e) Considerable new material was added in Chapter 6:

  (1) An expanded discussion of one-step lookahead policies and associated performance bounds in Section 6.3.1.

  (2) A discussion of aggregation methods and discretization of continuous-state problems (see Subsection 6.3.4).

  (3) A discussion of model predictive control and its relation to other suboptimal control methods (see Subsection 6.5.2).

  (4) An expanded treatment of open-loop feedback control and related methods based on a restricted structure (see Subsection 6.5.3).

I have also added a few exercises, and revised a few sections while preserving their essential content. Thanks are due to Haixia Lin, who worked out several exercises, and to Janey Yu, who reviewed some of the new sections and gave me valuable feedback.

Dimitri P. Bertsekas
http://web.mit.edu/dimitrib/www/home.html
Summer 2005

Preface to the Fourth Edition

The fourth edition contains a substantial amount of new material, particularly on approximate DP in Chapter 6. This chapter was thoroughly reorganized and rewritten, to bring it in line, both with the contents of Vol. II, whose latest edition appeared in 2012, and with recent developments, which have propelled approximate DP to the forefront of attention.

Some of the highlights of the revision of Chapter 6 are an increased emphasis on one-step and multistep lookahead methods, parametric approximation architectures, neural networks, rollout, and Monte Carlo tree search. Among other applications, these methods have been instrumental in the recent spectacular success of computer Go programs. The material on approximate DP also provides an introduction and some perspective for the more analytically oriented treatment of Vol. II.

The material of the chapters other than Chapter 6 has been reorganized and somewhat enriched, but not nearly as much. I have just added a few exercises and illustrative examples, and revised a few sections while preserving their essential content. The material on minimum variance control of autoregressive and moving average linear models (Sections 5.3, 6.1.4, and Appendix F in the 3rd edition) was eliminated in this edition, as over time it became disconnected from the remainder of the book. This material is now covered far more comprehensively in specialized textbooks. The material on adaptive control (Section 6.1 in the 3rd edition) has also been substantially reduced. A copy of the 3rd edition material that has been omitted from the 4th edition is posted at the book's web site

http://www.athenasc.com/dpbook.html

together with other instructional resources, such as my classroom slides and solutions to the exercises marked with the symbol www. The course material from several offerings of my class can be found at the MIT Open CourseWare (OCW) site:

https://ocw.mit.edu/index.htm

Links to a series of video lectures on approximate DP and related topics may be found at my website, which also contains my research papers on the subject.

Another change is this edition is that the chapter sequence has been reordered, so that the book is now naturally divided in two parts. Part I consists of Chapters 1-4 that are fundamental and ideally should be read as a group. Part II consists of Chapters 5, 6 and 7, each of which is terminal. These chapters can be read independently of each other, and in fact they may be attempted by some readers immediately after Chapter 1, with relatively little loss of continuity. The introductory Section 1.1 explains in more detail the new structure of the book.

Together with Vol. II, this volume provides an instructor with flexibility to follow several different pathways though the material. In my recent graduate offerings of the subject at MIT, I have covered most of Chapters 1 and 2, parts of Chapters 3 and 4, and most of Chapter 5, in the first half of a semester, and then spent the second half of the semester on approximate DP using Chapter 6 and Vol. II.

As always, many thanks are due to the students in my classes at MIT and elsewhere for stimulating interactions. I would like to single out Thomas Stahlbuhk, who has been very helpful with his comments. Thanks are also due to colleagues and collaborators, particularly John Tsitsiklis, Ben Van Roy, Mengdi Wang, and Huizhen Yu, for valuable interactions and suggestions for revision.

Dimitri P. Bertsekas
http://web.mit.edu/dimitrib/www/home.html
Winter 2017

On this 2nd Printing

In the 2nd printing of the 4th edition the approximate DP Chapter 6 has been updated and rewritten to bring it in line with the two reinforcement learning books written by the author in the interim [Ber19a], [Ber20a]. The remainder of the book was essentially left unchanged, and the total number of pages was also unchanged.

Dimitri P. Bertsekas
http://web.mit.edu/dimitrib/www/home.html
Summer 2020

1

The Dynamic Programming Algorithm

Contents

Life can only be understood going backwards,
but it must be lived going forwards.
Kierkegaard

1.1 INTRODUCTION

This book deals with situations where decisions are made in stages. The outcome of each decision may not be fully predictable but can be anticipated to some extent before the next decision is made. The objective is to minimize a certain cost over a given number of stages – a mathematical expression of what is considered an undesirable outcome.

A key aspect of such situations is that decisions cannot be viewed in isolation since one must balance the desire for low present cost with the undesirability of high future costs. The dynamic programming (DP) technique captures this tradeoff. At each stage, it ranks decisions based on the sum of the present cost and the expected future cost, assuming optimal decision making for subsequent stages.

There is a very broad variety of practical problems that can be treated by DP. In this book, we try to keep the main ideas uncluttered by irrelevant assumptions on problem structure. To this end, we formulate a broadly applicable model of stochastic optimal control of a dynamic system with perfect state observations over a finite number of stages (a finite horizon). This model will be the starting point for our development, and will occupy us for the first four of the seven chapters of this volume (which may be viewed as Part I of the book). The last three chapters (which may be viewed as Part II of the book) deal with related topics, and are terminal chapters for the purposes of this volume. In fact each of these chapters may be attempted by some readers immediately after Chapter 1, with relatively little loss of continuity. In summary, the seven chapters are structured as follows (see Fig. 1.1.1):

(1) The first chapter deals with the formulation of a general optimal control problem over a finite horizon, it demonstrates its broad applicability in deterministic and stochastic settings, and develops the DP algorithm for its solution.

(2) The second chapter considers the deterministic finite-state case of the problem. It explores the connections with the classical shortest path problem, and the special algorithms that this connection brings to bear.

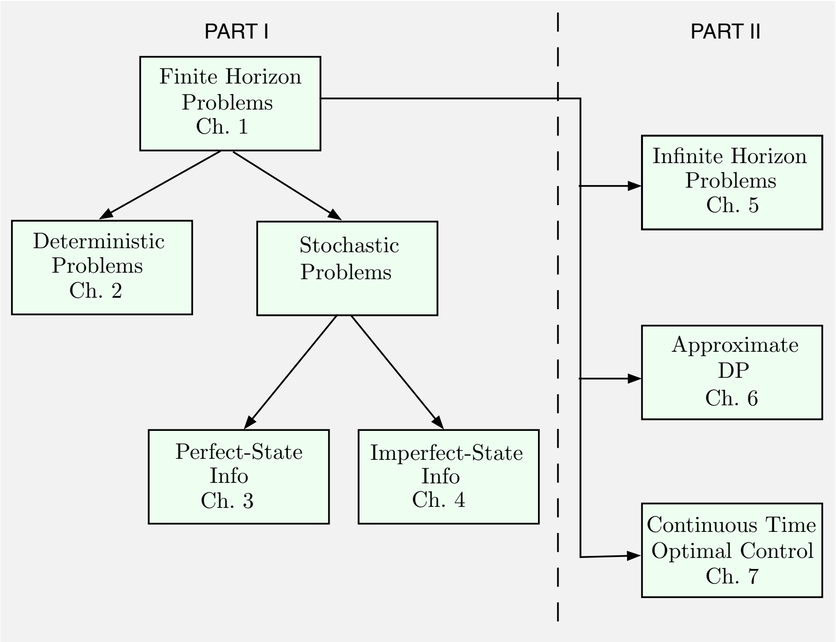

PART I
PART II
Finite Horizon Problems Ch. 1
Deterministic Problems Ch. 2
Stochastic Problems
Perfect-State Info Ch. 3
Imperfect-State Info Ch. 4
Infinite Horizon Problems Ch. 5
Approximate DP Ch. 6
Continuous Time Optimal Control Ch. 7

Figure 1.1.1 Illustration of the structure of the seven chapters of the present volume.

(3) The third chapter considers the stochastic general-state case of the problem, and illustrates various aspects of the solution process by means of some important applications.

(4) The fourth chapter also considers the stochastic general-state case, but contrary to the third chapter, it considers the situation where the exact state of the system is not observed perfectly by the decision maker/controller. This is a much harder problem, but conceptually it is closely related to the case of perfect state observation. Much of the chapter is devoted to explaining this important conceptual connection.

(5) The fifth chapter is an introduction to the theory of infinite horizon problems. It focuses on the relatively easy but important case of finite state problems. Volume II considers infinite horizon problems in greater generality and depth.

(6) The sixth chapter considers approximations to the exact DP solution method. This is a subject of great importance in practice, with a rich algorithmic methodology and very broad applications. We focus here primarily on finite horizon problems, so that this chapter can be read independently of Chapter 5. However, much of the discussion extends to infinite horizon problems, and on occasion we pause to indicate the

nature of the extension. We will consider approximate DP for infinite horizon problems in greater detail in Vol. II.

(7) The seventh chapter is a terminal chapter on deterministic optimal control in continuous space and time. It may be skipped without loss of continuity. Alternatively, it may be read immediately following Chapter 1. Among others, we emphasize here the methodological connections with DP and the analog of the DP algorithm in continuous time, which is the Hamilton-Jacobi-Bellman equation.

1.1.1 General Structure of Finite Horizon Optimal Control Problems

Our finite horizon model has two principal features: (1) a *discrete-time dynamic system*, and (2) a *cost function that is additive over time*. The dynamic system expresses the evolution of some variables, the system's "state," under the influence of decisions made at discrete instances of time. The system has the form

$$x_{k+1} = f_k(x_k, u_k, w_k), \qquad k = 0, 1, \ldots, N-1,$$

where

k indexes discrete time,

x_k is the state of the system and summarizes past information that is relevant for future optimization,

u_k is the control or decision variable to be selected at time k,

w_k is a random parameter (also called disturbance or noise depending on the context),

N is the horizon or number of times control is applied,

and f_k is a function that describes the system and in particular the mechanism by which the state is updated.

The cost function is additive in the sense that the cost incurred at time k, denoted by $g_k(x_k, u_k, w_k)$, accumulates over time. The total cost is

$$g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k),$$

where $g_N(x_N)$ is a terminal cost incurred at the end of the process. However, because of the presence of w_k, the cost is generally a random variable and cannot be meaningfully optimized. We therefore formulate the problem as an optimization of the *expected cost*

$$E\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right\},$$

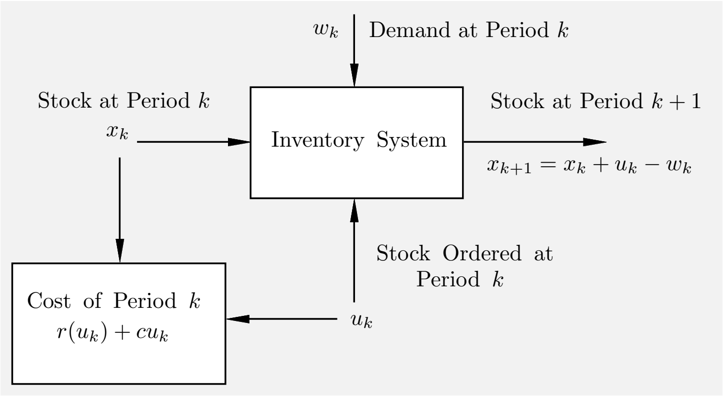


Figure 1.1.2 Inventory control example. At period k, the current stock (state) x_k, the stock ordered (control) u_k, and the demand (random disturbance) w_k determine the cost $r(x_k) + cu_k$ of period k and the stock $x_{k+1} = x_k + u_k - w_k$ at the next period.

where the expectation is with respect to the joint distribution of the random variables involved. The optimization is over the controls $u_0, u_1, \ldots, u_{N-1}$, but some qualification is needed here: each control u_k is selected with some knowledge of the current state x_k (either its exact value or some other related information).

A more precise definition of the terminology just used will be given shortly. We first provide some orientation by means of examples.

Example 1.1.1 (Inventory Control - Open-Loop and Closed-Loop Optimization)

Consider a problem of ordering a quantity of a certain item at each of N periods so as to (roughly) meet a stochastic demand, while minimizing the incurred expected cost. Let us denote

x_k stock available at the beginning of the kth period,

u_k stock ordered (and immediately delivered) at the beginning of the kth period,

w_k demand during the kth period with given probability distribution.

We assume that $w_0, w_1, \ldots, w_{N-1}$ are independent random variables, and that excess demand is backlogged and filled as soon as additional inventory becomes available. Thus, stock evolves according to the discrete-time equation

$$x_{k+1} = x_k + u_k - w_k,$$

where negative stock corresponds to backlogged demand (see Fig. 1.1.2).

The cost incurred in period k consists of two components:

(a) A cost $r(x_k)$ representing a penalty for either positive stock x_k (holding cost for excess inventory) or negative stock x_k (shortage cost for unfilled demand).

(b) The purchasing cost cu_k, where c is cost per unit ordered.

There is also a terminal cost $R(x_N)$ for being left with inventory x_N at the end of N periods. Thus, the total cost over N periods is

$$E\left\{R(x_N) + \sum_{k=0}^{N-1} \bigl(r(x_k) + cu_k\bigr)\right\}.$$

We want to minimize this cost by proper choice of the orders $u_0, \ldots, u_{N-1}$, subject to the natural constraint $u_k \geq 0$ for all k.

At this point we need to distinguish between *closed-loop* and *open-loop* minimization of the cost. In open-loop minimization we select all orders $u_0, \ldots, u_{N-1}$ at once at time 0, without waiting to see the subsequent demand levels. In closed-loop minimization we postpone placing the order u_k until the last possible moment (time k) when the current stock x_k will be known. The idea is that since there is no penalty for delaying the order u_k up to time k, we can take advantage of information that becomes available between times 0 and k (the demand and stock level in past periods).

Closed-loop optimization is of central importance in DP and is the type of optimization that we will consider almost exclusively in this book. Thus, in our basic formulation, decisions are made in stages while gathering information between stages that will be used to enhance the quality of the decisions. The effect of this on the structure of the resulting optimization problem is quite profound. In particular, in closed-loop inventory optimization we are not interested in finding optimal numerical values of the orders but rather we want to find an *optimal rule for selecting at each period k an order u_k for each possible value of stock x_k that can conceivably occur*. This is an "action versus strategy" distinction.

Mathematically, in closed-loop inventory optimization, we want to find a sequence of functions μ_k, $k = 0, \ldots, N-1$, mapping stock x_k into order u_k so as to minimize the expected cost. The meaning of μ_k is that, for each k and each possible value of x_k,

$$\mu_k(x_k) = \text{ amount that should be ordered at time } k \text{ if the stock is } x_k.$$

The sequence $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$ will be referred to as a *policy* or *control law*. For each π, the corresponding cost for a fixed initial stock x_0 is

$$J_\pi(x_0) = E\left\{R(x_N) + \sum_{k=0}^{N-1} \bigl(r(x_k) + c\mu_k(x_k)\bigr)\right\},$$

and we want to minimize $J_\pi(x_0)$ for a given x_0 over all π that satisfy the constraints of the problem. This is a typical DP problem. We will analyze this problem in various forms in subsequent sections. For example, we will show in Section 3.2 that for a reasonable choice of the cost function, the optimal ordering policy is of the form

$$\mu_k(x_k) = \begin{cases} S_k - x_k & \text{if } x_k < S_k, \\ 0 & \text{otherwise,} \end{cases}$$

where S_k is a suitable threshold level determined by the data of the problem. In other words, when stock falls below the threshold S_k, order just enough to bring stock up to S_k.

The preceding example illustrates the main ingredients of our formulation:

(a) A *discrete-time system* of the form

$$x_{k+1} = f_k(x_k, u_k, w_k),$$

where f_k is some function; for example in the inventory case, we have $f_k(x_k, u_k, w_k) = x_k + u_k - w_k$.

(b) *Independent random parameters* w_k. This will be generalized by allowing the probability distribution of w_k to depend on x_k and u_k; in the context of the inventory example, we can think of a case where the level of demand w_k is affected by the current stock level x_k.

(c) A *control constraint*; in the example, we have $u_k \geq 0$. In general, the constraint set will depend on x_k and the time index k, that is, $u_k \in U_k(x_k)$. To see how constraints dependent on x_k can arise in the inventory context, think of a situation where there is an upper bound B on the level of stock that can be accommodated, so $u_k \leq B - x_k$.

(d) An *additive cost* of the form

$$E\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right\},$$

where g_k are some functions; in the inventory example, we have $g_N(x_N) = R(x_N)$ and $g_k(x_k, u_k, w_k) = r(x_k) + cu_k$.

(e) *Optimization over (closed-loop) policies*, i.e., rules for choosing u_k for each k and each possible value of x_k.

We next consider the important special case where in addition to discrete time, the problem has a discrete state structure.

1.1.2 Discrete-State and Finite-State Problems

In the preceding example, the state x_k was a continuous real variable, and it is easy to think of multidimensional generalizations where the state is an n-dimensional vector of real variables. It is also possible, however, that the state takes values from a discrete set, such as the integers.

A version of the inventory problem where a discrete viewpoint is more natural arises when stock is measured in whole units (such as cars), each of which is a significant fraction of x_k, u_k, or w_k. It is more appropriate then to take as state space the set of all integers rather than the set of real numbers. The form of the system equation and the cost per period will, of course, stay the same.

Generally, there are many situations where the state is naturally discrete and there is no continuous counterpart of the problem. Such situations are often conveniently specified in terms of the probabilities of transition between the states. What we need to know is $p_{ij}(u,k)$, which is the probability at time k that the next state will be j, given that the current state is i, and the control selected is u, i.e.,

$$p_{ij}(u,k) = P\big\{x_{k+1} = j \mid x_k = i, u_k = u\big\}.$$

This type of state transition can alternatively be described in terms of the discrete-time system equation

$$x_{k+1} = w_k,$$

where the probability distribution of the random parameter w_k is

$$P\big\{w_k = j \mid x_k = i, u_k = u\big\} = p_{ij}(u,k).$$

Conversely, given a discrete-state system in the form

$$x_{k+1} = f_k(x_k, u_k, w_k),$$

together with the probability distribution $P_k(w_k \mid x_k, u_k)$ of w_k, we can provide an equivalent transition probability description. The corresponding transition probabilities are given by

$$p_{ij}(u,k) = P_k\big\{W_k(i,u,j) \mid x_k = i, u_k = u\big\},$$

where $W(i,u,j)$ is the set

$$W_k(i,u,j) = \big\{w \mid j = f_k(i,u,w)\big\}.$$

Thus a discrete-state system can equivalently be described in terms of a difference equation or in terms of transition probabilities. Depending on the given problem, it may be notationally or mathematically more convenient to use one description over the other.

The following examples illustrate discrete-state problems. The first example involves a *deterministic* problem, i.e., a problem where there is no stochastic uncertainty. In such a problem, when a control is chosen at a given state, the next state is fully determined; i.e., for any state i, control u, and time k, the transition probability $p_{ij}(u,k)$ is equal to 1 for a single state j, and it is 0 for all other candidate next states. The other three examples involve stochastic problems, where the next state resulting from a given choice of control at a given state cannot be determined a priori.

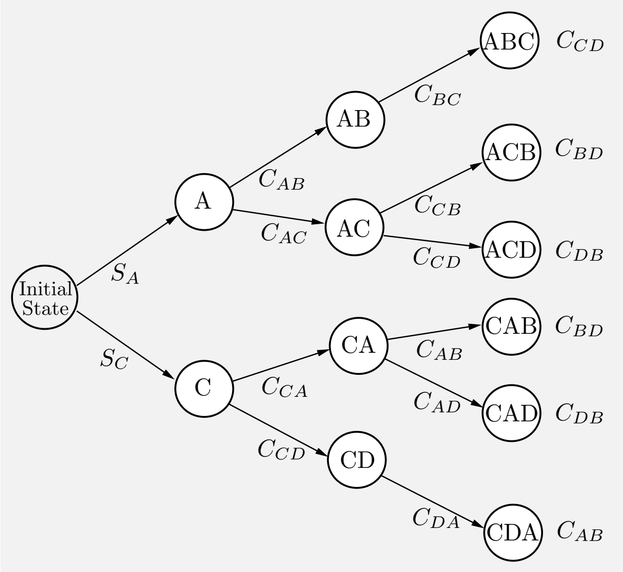


Figure 1.1.3 The transition graph of the deterministic scheduling problem of Example 1.1.2. Each arc of the graph corresponds to a decision leading from some state (the start node of the arc) to some other state (the end node of the arc). The corresponding cost is shown next to the arc. The cost of the last operation is shown as a terminal cost next to the terminal nodes of the graph.

Example 1.1.2 (A Deterministic Scheduling Problem)

Suppose that to produce a certain product, four operations must be performed on a certain machine. The operations are denoted by A, B, C, and D. We assume that operation B can be performed only after operation A has been performed, and operation D can be performed only after operation C has been performed. (Thus the sequence CDAB is allowable but the sequence CDBA is not.) The setup cost C_{mn} for passing from any operation m to any other operation n is given. There is also an initial startup cost S_A or S_C for starting with operation A or C, respectively (cf. Fig. 1.1.3). The cost of a sequence is the sum of the setup costs associated with it; for example, the operation sequence ACDB has cost

$$S_A + C_{AC} + C_{CD} + C_{DB}.$$

We can view this problem as a sequence of three decisions, namely the choice of the first three operations to be performed (the last operation is determined from the preceding three). It is appropriate to consider as state the set of operations already performed, the initial state being an artificial state corresponding to the beginning of the decision process. The possible state transitions corresponding to the possible states and decisions for this problem are shown in Fig. 1.1.3. Here the problem is deterministic, i.e., at a given state, each choice of control leads to a uniquely determined state. For example, at state AC the decision to perform operation D leads to state ACD with certainty, and has cost C_{CD}. Deterministic problems with a finite number of states can be conveniently represented in terms of transition graphs such as the one of Fig. 1.1.3. The optimal solution corresponds to the path that starts at the initial state and ends at some state at the terminal time and has minimum sum of arc costs plus the terminal cost. We will study systematically problems of this type in Chapter 2.

Example 1.1.3 (Machine Replacement)

Consider a problem of operating efficiently over N time periods a machine that can be in any one of n states, denoted $1, 2, \ldots, n$. We denote by $g(i)$ the operating cost per period when the machine is in state i, and we assume that

$$g(1) \leq g(2) \leq \cdots \leq g(n).$$

The implication here is that state i is better than state $i+1$, and state 1 corresponds to a machine in best condition.

During a period of operation, the state of the machine can become worse or it may stay unchanged. We thus assume that the transition probabilities

$$p_{ij} = P\{\text{next state will be } j \mid \text{current state is } i\}$$

satisfy

$$p_{ij} = 0 \qquad \text{if } j < i.$$

We assume that at the start of each period we know the state of the machine and we must choose one of the following two options:

(a) Let the machine operate one more period in the state it currently is.

(b) Repair the machine and bring it to the best state 1 at a cost R.

We assume that the machine, once repaired, is guaranteed to stay in state 1 for one period. In subsequent periods, it may deteriorate to states $j > 1$ according to the transition probabilities p_{1j}.

Thus the objective here is to decide on the level of deterioration (state) at which it is worth paying the cost of machine repair, thereby obtaining the benefit of smaller future operating costs. Note that the decision should also be affected by the period we are in. For example, we would be less inclined to repair the machine when there are few periods left.

The system evolution for this problem can be described by the graphs of Fig. 1.1.4. These graphs depict the transition probabilities between various pairs of states for each value of the control and are known as *transition probability graphs* or simply *transition graphs*. Note that there is a different graph for each of the two controls.

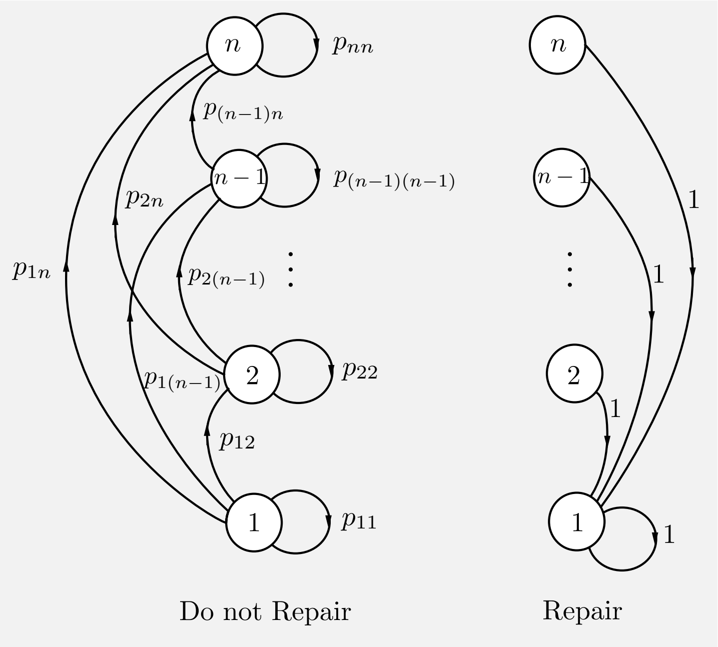


Figure 1.1.4 Machine replacement example. Transition probability graphs for each of the two possible controls (repair or not repair). At each stage and state i, the cost of repairing is $R + g(1)$, and the cost of not repairing is $g(i)$. The terminal cost is 0.

Example 1.1.4 (Control of a Queue)

Consider a queueing system with room for n customers operating over N time periods. We assume that service of a customer can start (end) only at the beginning (end) of the period and that the system can serve only one customer at a time. The probability p_m of m customer arrivals during a period is given, and the numbers of arrivals in two different periods are independent. Customers finding the system full depart without attempting to enter later. The system offers two kinds of service, *fast* and *slow*, with cost per period c_f and c_s, respectively. Service can be switched between fast and slow at the beginning of each period. With fast (slow) service, a customer in service at the beginning of a period will terminate service at the end of the period with probability q_f (respectively, q_s) independently of the number of periods the customer has been in service and the number of customers in the system ($q_f > q_s$). There is a cost $r(i)$ for each period for which there are i customers in the system. There is also a terminal cost $R(i)$ for i customers left in the system at the end of the last period.

The problem is to choose, at each period, the type of service as a function of the number of customers in the system so as to minimize the expected total cost over N periods. One expects that when there is a large number of customers i in queue, it is better to use the fast service, and the question is to find the values of i for which this is true.

Here it is appropriate to take as state the number i of customers in the system at the start of a period and as control the type of service provided. Then, the cost per period is $r(i)$ plus c_f or c_s depending on whether fast or slow service is provided. We derive the transition probabilities of the system.

When the system is empty at the start of the period, the probability that the next state is j is independent of the type of service provided. It equals the given probability of j customer arrivals when $j < n$,

$$p_{0j}(u_f) = p_{0j}(u_s) = p_j, \qquad j = 0, 1, \ldots, n-1,$$

and it equals the probability of n or more customer arrivals when $j = n$,

$$p_{0n}(u_f) = p_{0n}(u_s) = \sum_{m=n}^{\infty} p_m.$$

When there is at least one customer in the system ($i > 0$), we have

$$p_{ij}(u_f) = 0, \qquad \text{if } j < i - 1,$$

$$p_{ij}(u_f) = q_f p_0, \qquad \text{if } j = i - 1,$$

$$\begin{aligned} p_{ij}(u_f) &= P\{j - i + 1 \text{ arrivals, service completed}\} \\ &\quad + P\{j - i \text{ arrivals, service not completed}\} \\ &= q_f p_{j-i+1} + (1 - q_f) p_{j-i}, \qquad \text{if } i - 1 < j < n - 1, \end{aligned}$$

$$p_{i(n-1)}(u_f) = q_f \sum_{m=n-i}^{\infty} p_m + (1 - q_f) p_{n-1-i},$$

$$p_{in}(u_f) = (1 - q_f) \sum_{m=n-i}^{\infty} p_m.$$

The transition probabilities when slow service is provided are also given by these formulas with u_f and q_f replaced by u_s and q_s, respectively.

Example 1.1.5 (Optimizing a Chess Match Strategy)

A player is about to play a two-game chess match with an opponent, and wants to maximize his winning chances. Each game can have one of two outcomes:

(a) A win by one of the players (1 point for the winner and 0 for the loser).

(b) A draw (1/2 point for each of the two players).

If the score is tied at 1-1 at the end of the two games, the match goes into sudden-death mode, whereby the players continue to play until the first time one of them wins a game (and the match). The player has two playing styles and he can choose one of the two at will in each game, independently of the style he chose in previous games.

(1) *Timid play* with which he draws with probability $p_d > 0$, and he loses with probability $(1 - p_d)$.

(2) *Bold play* with which he wins with probability p_w, and he loses with probability $(1 - p_w)$.

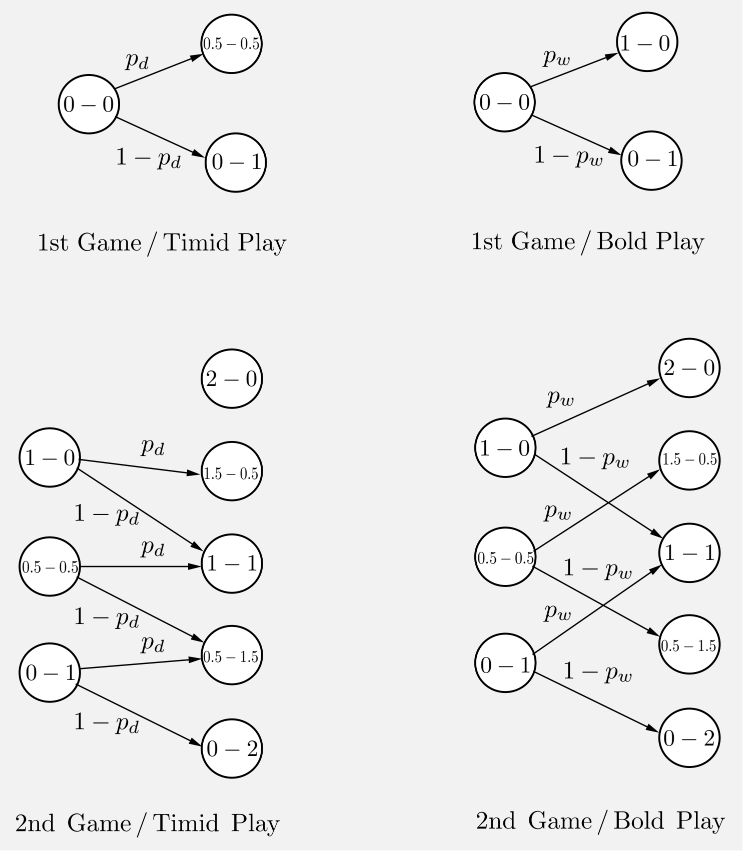


Figure 1.1.5 Chess match problem for Example 1.1.5. Transition probability graphs for each of the two possible controls (timid or bold play). Note here that the state space is not the same at each stage. The terminal cost is -1 at the winning final scores 2-0 and 1.5-0.5, 0 at the losing final scores 0-2 and 0.5-1.5, and $-p_w$ at the tied score 1-1.

Thus, in a given game, timid play never wins, while bold play never draws. The player wants to find a style selection strategy that maximizes his probability of winning the match. Note that once the match gets into sudden death, the player should play bold, since with timid play he can at best prolong the sudden death play, while running the risk of losing. Therefore, there are only two decisions for the player to make, the selection of the playing strategy in the first two games.

We can model the problem as one with two stages, and with states the possible scores at the start of each of the first two stages (games), as shown in Fig. 1.1.5. The initial state is the initial score 0-0. The transition probabilities for each of the two different controls (playing styles) are also shown in Fig. 1.1.5. There is a cost at the terminal states: a cost of -1 at the winning scores 2-0 and 1.5-0.5, a cost of 0 at the losing scores 0-2 and 0.5-1.5, and a cost of $-p_w$ at the tied score 1-1 (since the probability of winning in sudden death is p_w). Note that to maximize the probability P of winning the match, we must minimize $-P$.

This problem has an interesting feature. One would think that if $p_w < 1/2$, the player would have a less than 50-50 chance of winning the match, even with optimal play, since his probability of losing is greater than his probability of winning any one game, regardless of his playing style. This is not so, however, because the player can adapt his playing style to the current score, but his opponent does not have that option. In other words, the player can use a closed-loop strategy, and it will be seen later that with optimal play, as determined by the DP algorithm, he has a better than 50-50 chance of winning the match provided p_w is higher than a threshold value $\overline{p}$, which, depending on the value of p_d, may satisfy $\overline{p} < 1/2$.

1.2 THE BASIC PROBLEM

We now formulate a general problem of decision under stochastic uncertainty over a finite number of stages. This problem, which we call *basic*, is central in this book. We will discuss solution methods based on DP in the first five chapters, and we will extend our analysis to versions of this problem involving an infinite number of stages in Chapter 5 and in Vol. II of this work.

The basic problem is very general. In particular, we will not require that the state, control, or random parameter take a finite number of values or belong to a space of n-dimensional vectors. A surprising aspect of DP is that its applicability depends very little on the nature of the state, control, and random parameter spaces. For this reason it is convenient to proceed without any assumptions on the structure of these spaces; indeed such assumptions would become a serious impediment later.

Basic Problem

We are given a discrete-time dynamic system

$$x_{k+1} = f_k(x_k, u_k, w_k), \qquad k = 0, 1, \ldots, N-1,$$

where the state x_k is an element of a space S_k, the control u_k is an element of a space C_k, and the random "disturbance" w_k is an element of a space D_k.

The control u_k is constrained to take values in a given nonempty subset $U(x_k) \subset C_k$, which depends on the current state x_k; i.e., $u_k \in U_k(x_k)$ for all $x_k \in S_k$ and k.

The random disturbance w_k is characterized by a probability distribution $P_k(\cdot \mid x_k, u_k)$ that may depend explicitly on x_k and u_k but not on values of prior disturbances $w_{k-1}, \ldots, w_0$.

We consider the class of policies (also called control laws) that consist of a sequence of functions

$$\pi = \{\mu_0, \ldots, \mu_{N-1}\},$$

where μ_k maps states x_k into controls $u_k = \mu_k(x_k)$ and is such that $\mu_k(x_k) \in U_k(x_k)$ for all $x_k \in S_k$. Such policies will be called *admissible*.

Given an initial state x_0 and an admissible policy $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$, the states x_k and disturbances w_k are random variables with distributions defined through the system equation

$$x_{k+1} = f_k\big(x_k, \mu_k(x_k), w_k\big), \qquad k = 0, 1, \ldots, N-1. \tag{1.1}$$

Thus, for given functions g_k, $k = 0, 1, \ldots, N$, the expected cost of π starting at x_0 is

$$J_\pi(x_0) = E\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k\big(x_k, \mu_k(x_k), w_k\big) \right\}$$

where the expectation is taken over the random variables w_k and x_k. An optimal policy π^* is one that minimizes this cost; i.e.,

$$J_{\pi^*}(x_0) = \min_{\pi \in \Pi} J_\pi(x_0),$$

where Π is the set of all admissible policies.

Note that the optimal policy π^* is associated with a fixed initial state x_0. However, an interesting aspect of the basic problem and of DP is that it is typically possible to find a policy π^* that is simultaneously optimal for all initial states.

The optimal cost depends on x_0 and is denoted by $J^*(x_0)$; i.e.,

$$J^*(x_0) = \min_{\pi \in \Pi} J_\pi(x_0).$$

It is useful to view J^* as a function that assigns to each initial state x_0 the optimal cost $J^*(x_0)$ and call it the *optimal cost function* or *optimal value function*.†

The Role and Value of Information

We noted earlier the distinction between open-loop minimization, where we select all controls $u_0, \ldots, u_{N-1}$ at once at time 0, and closed-loop minimization, where we select a policy $\{\mu_0, \ldots, \mu_{N-1}\}$ that applies the control $\mu_k(x_k)$ at time k with knowledge of the current state x_k (see Fig. 1.2.1). With closed-loop policies, it is possible to achieve lower cost, essentially by taking advantage of the extra information (the knowledge of the current state). The reduction in cost may be called the *value of the information* and can be significant indeed. If the information is not available, the controller cannot adapt appropriately to unexpected values of the state, and as a result the cost can be adversely affected. For example, in the inventory control example of the preceding section, the information that becomes available at the beginning of each period k is the inventory stock x_k. Clearly, this is important information to the inventory manager, who will want to adjust the amount u_k to be purchased depending on whether the current stock x_k is running high or low.

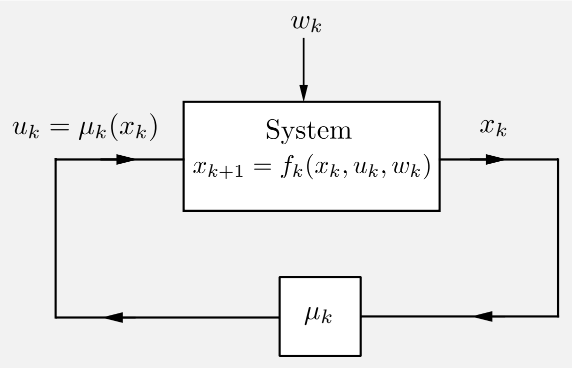


Figure 1.2.1 Information gathering in the basic problem. At each time k the controller observes the current state x_k and applies a control $u_k = \mu_k(x_k)$ that depends on that state.

† For the benefit of the mathematically oriented reader we note that in the preceding equation, "min" denotes the greatest lower bound (or infimum) of the set of numbers $\{J_\pi(x_0) \mid \pi \in \Pi\}$. A notation more in line with normal mathematical usage would be to write $J^*(x_0) = \inf_{\pi \in \Pi} J_\pi(x_0)$. However (as discussed in Appendix B), we find it convenient to use "min" in place of "inf" even when the infimum is not attained. It is less distracting, and it will not lead to any confusion.

Example 1.2.1

To illustrate the benefits of the proper use of information, let us consider the chess match example of the preceding section (Example 1.1.5). There, a player can select timid play (probabilities p_d and $1 - p_d$ for a draw and a loss, respectively) or bold play (probabilities p_w and $1 - p_w$ for a win and a loss, respectively) in each of the two games of the match. Suppose the player chooses a policy of playing timid if and only if he is ahead in the score, as illustrated in Fig. 1.2.2; we will see in the next section that this policy is optimal, assuming $p_d > p_w$. Then after the first game (in which he plays bold), the score is 1-0 with probability p_w and 0-1 with probability $1 - p_w$. In the second game, he plays timid in the former case and bold in the latter case. Thus after two games, the probability of a match win is $p_w p_d$, the probability of a match loss is $(1 - p_w)^2$, and the probability of a tied score is $p_w(1-p_d) + (1-p_w)p_w$, in which case he has a probability p_w of winning the subsequent sudden-death game. Thus the probability of winning the match with the given strategy is

$$p_w p_d + p_w\big(p_w(1 - p_d) + (1 - p_w)p_w\big),$$

which, with some rearrangement, gives

$$\text{Probability of a match win } = p_w^2(2 - p_w) + p_w(1 - p_w)p_d. \tag{1.2}$$

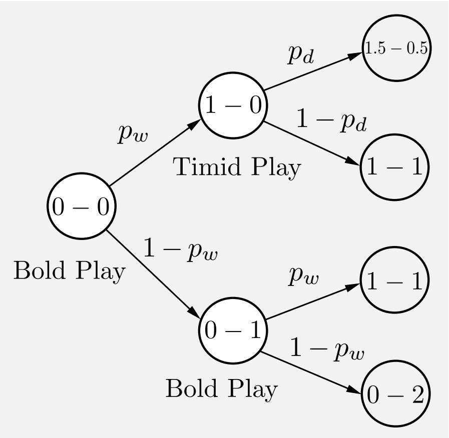


Figure 1.2.2 Illustration of the policy used in Example 1.2.1 to obtain a greater than 50-50 chance of winning the chess match and associated transition probabilities. The player chooses a policy of playing timid if and only if he is ahead in the score.

Suppose now that $p_w < 1/2$. Then the player has a greater probability of losing than winning any one game, regardless of the type of play he uses.

From this we can infer that no open-loop strategy can give the player a greater than 50-50 chance of winning the match. Yet from Eq. (1.2) it can be seen that with the closed-loop strategy of playing timid if and only if the player is ahead in the score, the chance of a match win can be greater than 50-50, provided that p_w is close enough to 1/2 and p_d is close enough to 1. As an example, for $p_w = 0.45$ and $p_d = 0.9$, Eq. (1.2) gives a match win probability of roughly 0.53.

To calculate the value of information, let us consider the four open-loop policies, whereby we decide on the type of play to be used without waiting to see the result of the first game. These are:

(1) Play timid in both games; this has a probability $p_d^2 p_w$ of winning the match.

(2) Play bold in both games; this has a probability $p_w^2 + 2p_w^2(1-p_w) = p_w^2(3-2p_w)$ of winning the match.

(3) Play bold in the first game and timid in the second game; this has a probability $p_w p_d + p_w^2(1-p_d)$ of winning the match.

(4) Play timid in the first game and bold in the second game; this also has a probability $p_w p_d + p_w^2(1-p_d)$ of winning the match.

The first policy is always dominated by the others, and the optimal open-loop probability of winning the match is

$$\begin{aligned}\text{Open-loop probability of win} &= \max\big(p_w^2(3-2p_w),\ p_w p_d + p_w^2(1-p_d)\big)\\ &= p_w^2 + p_w(1-p_w)\max(2p_w, p_d).\end{aligned} \tag{1.3}$$

Thus if $p_d > 2p_w$, we see that the optimal open-loop policy is to play timid in one of the two games and play bold in the other, while if $p_d \leq 2p_w$, it is optimal to play bold in both games.

As an example, for $p_w = 0.45$ and $p_d = 0.9$, Eq. (1.3) gives an optimal open-loop match win probability of roughly 0.425. Thus, the value of the information (the outcome of the first game) is the difference of the optimal closed-loop and open-loop values, which is approximately $0.53 - 0.425 = 0.105$. More generally, by subtracting Eqs. (1.2) and (1.3), we see that

$$\begin{aligned}\text{Value of information } &= p_w^2(2-p_w) + p_w(1-p_w)p_d\\ &\quad - p_w^2 - p_w(1-p_w)\max(2p_w, p_d)\\ &= p_w(1-p_w)\min(p_w, p_d - p_w).\end{aligned}$$

We note, however, that whereas availability of the state information cannot hurt, it may not result in an advantage either. For instance, in deterministic problems, where no random disturbances are present, one can predict the future states given the initial state and the sequence of controls. Thus, optimization over all sequences $\{u_0, u_1, \ldots, u_{N-1}\}$ of controls leads to the same optimal cost as optimization over all admissible policies. The same can be true even in some stochastic problems (see for example Exercise 1.27). This brings up a related issue. Assuming no information is forgotten, the controller actually knows the prior states and controls $x_0, u_0, \ldots, x_{k-1}, u_{k-1}$ as well as the current state x_k. Therefore, the question arises whether policies that use the entire system history can be superior to policies that use just the current state. The answer turns out to be negative although the proof is technically complicated (see, e.g., [BeS78]). The intuitive reason is that, for a given time k and state x_k, all future expected costs depend explicitly just on x_k and not on prior history.

Encoding Risk in the Cost Function

As mentioned above, an important characteristic of stochastic problems is the possibility of using information with advantage. Another distinguishing characteristic is the need to take into account *risk* in the problem formulation. For example, in a typical investment problem one is not only interested in the expected profit of the investment decision, but also in its variance: given a choice between two investments with nearly equal expected profit and markedly different variance, most investors would prefer the investment with smaller variance. This indicates that expected value of cost or reward need not be the most appropriate yardstick for expressing a decision maker's preference between decisions.

As a more dramatic example of the need to take risk into account when formulating optimization problems under uncertainty, consider the so-called St. Petersburg paradox. Here, a person is offered the opportunity of paying x dollars in exchange for participation in the following game: a fair coin is flipped sequentially and the person is paid 2^k dollars, where k is the number of times heads have come up before tails come up for the first time. The decision that the person must make is whether to accept or reject participation in the game. Now if he accepts, his expected profit from the game is

$$\sum_{k=0}^{\infty} \frac{1}{2^{k+1}} \cdot 2^k - x = \infty,$$

so if his acceptance criterion is based on maximization of expected profit, he is willing to pay any amount x to enter the game. This, however, is in strong disagreement with observed behavior, due to the risk element involved in entering the game, and shows that a different formulation of the problem is needed. The formulation of problems of decision under uncertainty so that risk is properly taken into account is a deep subject with an interesting theory. An introduction to this theory is given in Appendix F. It is shown in particular that minimization of expected cost is appropriate under reasonable assumptions, provided the cost function is suitably chosen so that it properly encodes the risk preferences of the decision maker.

1.3 THE DYNAMIC PROGRAMMING ALGORITHM

The DP algorithm rests on a very simple idea, the *principle of optimality*. The name is due to Bellman, who contributed a great deal to the popularization of DP and to its transformation into a systematic tool. Roughly, the principle of optimality states the following rather obvious fact.

Principle of Optimality

Let $\pi^* = \{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$ be an optimal policy for the basic problem, and assume that when using π^*, a given state x_i occurs at time i with positive probability. Consider the subproblem whereby we are at x_i at time i and wish to minimize the "cost-to-go" from time i to time N

$$E\left\{ g_N(x_N) + \sum_{k=i}^{N-1} g_k\big(x_k, \mu_k(x_k), w_k\big) \right\}.$$

Then the truncated policy $\{\mu_i^*, \mu_{i+1}^*, \ldots, \mu_{N-1}^*\}$ is optimal for this subproblem.

The intuitive justification of the principle of optimality is very simple. If the truncated policy $\{\mu_i^*, \mu_{i+1}^*, \ldots, \mu_{N-1}^*\}$ were not optimal as stated, we would be able to reduce the cost further by switching to an optimal policy for the subproblem once we reach x_i. For an auto travel analogy, suppose that the fastest route from Los Angeles to Boston passes through Chicago. The principle of optimality translates to the obvious fact that the Chicago to Boston portion of the route is also the fastest route for a trip that starts from Chicago and ends in Boston.

The principle of optimality suggests that an optimal policy can be constructed in piecemeal fashion, first constructing an optimal policy for the "tail subproblem" involving the last stage, then extending the optimal policy to the "tail subproblem" involving the last two stages, and continuing in this manner until an optimal policy for the entire problem is constructed. The DP algorithm is based on this idea: it proceeds sequentially, by solving all the tail subproblems of a given time length, using the solution of the tail subproblems of shorter time length. We introduce the algorithm with two examples, one deterministic and one stochastic.

The DP Algorithm for a Deterministic Scheduling Example

Let us consider the scheduling Example 1.1.2, and let us apply the principle of optimality to calculate the optimal schedule. We have to schedule optimally the four operations A, B, C, and D. The numerical values of the transition and setup costs are shown in Fig. 1.3.1 next to the corresponding arcs.

According to the principle of optimality, the “tail” portion of an optimal schedule must be optimal. For example, suppose that the optimal schedule is CABD. Then, having scheduled first C and then A, it must be optimal to complete the schedule with BD rather than with DB. With this in mind, we solve all possible tail subproblems of length two, then all tail subproblems of length three, and finally the original problem that has length four (the subproblems of length one are of course trivial because there is only one operation that is as yet unscheduled). As we will see shortly, the tail subproblems of length $k+1$ are easily solved once we have solved the tail subproblems of length k, and this is the essence of the DP technique.

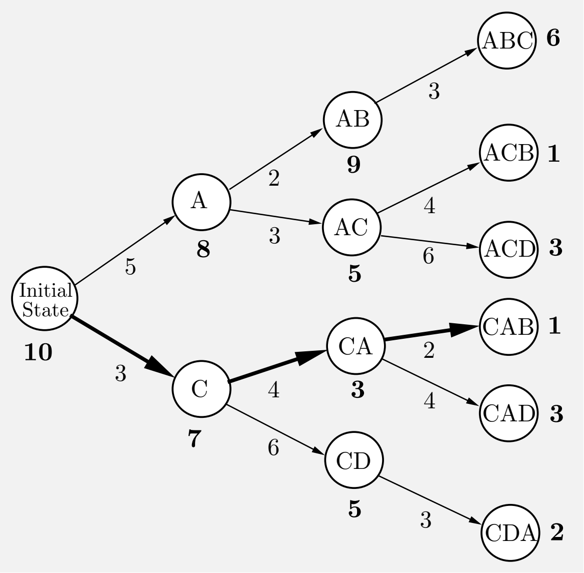


Figure 1.3.1 Transition graph of the deterministic scheduling problem, with the cost of each decision shown next to the corresponding arc. Next to each node/state we show the cost to optimally complete the schedule starting from that state. This is the optimal cost of the corresponding tail subproblem (cf. the principle of optimality). The optimal cost for the original problem is equal to 10, as shown next to the initial state. The optimal schedule corresponds to the thick-line arcs.

Tail Subproblems of Length 2: These subproblems are the ones that involve two unscheduled operations and correspond to the states AB, AC, CA, and CD (see Fig. 1.3.1)

State AB: Here it is only possible to schedule operation C as the next operation, so the optimal cost of this subproblem is 9 (the cost of scheduling C after B, which is 3, plus the cost of scheduling D after C, which is 6).

State AC: Here the possibilities are to (a) schedule operation B and then D, which has cost 5, or (b) schedule operation D and then B, which has cost 9. The first possibility is optimal, and the corresponding cost of the tail subproblem is 5, as shown next to node AC in Fig. 1.3.1.

State CA: Here the possibilities are to (a) schedule operation B and then D, which has cost 3, or (b) schedule operation D and then B, which has cost 7. The first possibility is optimal, and the corresponding cost of the tail subproblem is 3, as shown next to node CA in Fig. 1.3.1.

State CD: Here it is only possible to schedule operation A as the next operation, so the optimal cost of this subproblem is 5.

Tail Subproblems of Length 3: These subproblems can now be solved using the optimal costs of the subproblems of length 2.

State A: Here the possibilities are to (a) schedule next operation B (cost 2) and then solve optimally the corresponding subproblem of length 2 (cost 9, as computed earlier), a total cost of 11, or (b) schedule next operation C (cost 3) and then solve optimally the corresponding subproblem of length 2 (cost 5, as computed earlier), a total cost of 8. The second possibility is optimal, and the corresponding cost of the tail subproblem is 8, as shown next to node A in Fig. 1.3.1.

State C: Here the possibilities are to (a) schedule next operation A (cost 4) and then solve optimally the corresponding subproblem of length 2 (cost 3, as computed earlier), a total cost of 7, or (b) schedule next operation D (cost 6) and then solve optimally the corresponding subproblem of length 2 (cost 5, as computed earlier), a total cost of 11. The first possibility is optimal, and the corresponding cost of the tail subproblem is 7, as shown next to node A in Fig. 1.3.1.

Original Problem of Length 4: The possibilities here are (a) start with operation A (cost 5) and then solve optimally the corresponding subproblem of length 3 (cost 8, as computed earlier), a total cost of 13, or (b) start with operation C (cost 3) and then solve optimally the corresponding subproblem of length 3 (cost 7, as computed earlier), a total cost of 10. The second possibility is optimal, and the corresponding optimal cost is 10, as shown next to the initial state node in Fig. 1.3.1.

Note that having computed the optimal cost of the original problem through the solution of all the tail subproblems, we can construct the optimal schedule by starting at the initial node and proceeding forward, each time choosing the operation that starts the optimal schedule for the corresponding tail subproblem. In this way, by inspection of the graph and the computational results of Fig. 1.3.1, we determine that CABD is the optimal schedule.

The DP Algorithm for the Inventory Control Example

Consider the inventory control Example 1.1.1. Similar to the solution of the preceding deterministic scheduling problem, we calculate sequentially the optimal costs of all the tail subproblems, going from shorter to longer problems. The only difference is that the optimal costs are computed as expected values, since the problem here is stochastic.

Tail Subproblems of Length 1: Assume that at the beginning of period $N-1$ the stock is x_{N-1}. Clearly, no matter what happened in the past, the inventory manager should order the amount of inventory that minimizes over $u_{N-1} \geq 0$ the sum of the ordering cost and the expected terminal holding/shortage cost. Thus, he should minimize over u_{N-1} the sum $cu_{N-1} + E\{R(x_N)\}$, which can be written as

$$cu_{N-1} + \underset{w_{N-1}}{E}\big\{R(x_{N-1} + u_{N-1} - w_{N-1})\big\}.$$

Adding the holding/shortage cost of period $N-1$, we see that the optimal cost for the last period (plus the terminal cost) is given by

$$\begin{aligned} J_{N-1}(x_{N-1}) = r(x_{N-1})& \\ &+ \min_{u_{N-1}\geq 0}\left[cu_{N-1} + \underset{w_{N-1}}{E}\big\{R(x_{N-1} + u_{N-1} - w_{N-1})\big\}\right]. \end{aligned}$$

Naturally, J_{N-1} is a function of the stock x_{N-1}. It is calculated either analytically or numerically (in which case a table is used for computer storage of the function J_{N-1}). In the process of calculating J_{N-1}, we obtain the optimal inventory policy $\mu^*_{N-1}(x_{N-1})$ for the last period: $\mu^*_{N-1}(x_{N-1})$ is the value of u_{N-1} that minimizes the right-hand side of the preceding equation for a given value of x_{N-1}.

Tail Subproblems of Length 2: Assume that at the beginning of period $N-2$ the stock is x_{N-2}. It is clear that the inventory manager should order the amount of inventory that minimizes not just the expected cost of period $N-2$ but rather the

$$\begin{aligned} &(\text{expected cost of period } N-2) + (\text{expected cost of period } N-1, \\ &\qquad\qquad \text{given that an optimal policy will be used at period } N-1), \end{aligned}$$

which is equal to

$$r(x_{N-2}) + cu_{N-2} + E\big\{J_{N-1}(x_{N-1})\big\}.$$

Using the system equation $x_{N-1} = x_{N-2} + u_{N-2} - w_{N-2}$, the last term is also written as $J_{N-1}(x_{N-2} + u_{N-2} - w_{N-2})$.

Thus the optimal cost for the last two periods given that we are at state x_{N-2}, denoted $J_{N-2}(x_{N-2})$, is given by

$$\begin{aligned} J_{N-2}(x_{N-2}) = r(x_{N-2}) & \\ & + \min_{u_{N-2}\geq 0} \left[c u_{N-2} + \mathop{E}_{w_{N-2}} \big\{ J_{N-1}(x_{N-2} + u_{N-2} - w_{N-2}) \big\} \right] \end{aligned}$$

Again $J_{N-2}(x_{N-2})$ is calculated for every x_{N-2}. At the same time, the optimal policy $\mu^*_{N-2}(x_{N-2})$ is also computed.

Tail Subproblems of Length $N-k$: Similarly, we have that at period k, when the stock is x_k, the inventory manager should order u_k to minimize

(expected cost of period k) + (expected cost of periods $k+1, \ldots, N-1$,
given that an optimal policy will be used for these periods).

By denoting by $J_k(x_k)$ the optimal cost, we have

$$J_k(x_k) = r(x_k) + \min_{u_k \geq 0} \left[c u_k + \mathop{E}_{w_k} \big\{ J_{k+1}(x_k + u_k - w_k) \big\} \right], \tag{1.4}$$

which is actually the DP equation for this problem.

The functions $J_k(x_k)$ denote the optimal expected cost for the tail subproblem that starts at period k with initial inventory x_k. These functions are computed recursively backward in time, starting at period $N-1$ and ending at period 0. The value $J_0(x_0)$ is the optimal expected cost when the initial stock at time 0 is x_0. During the calculations, the optimal policy is simultaneously computed from the minimization in the right-hand side of Eq. (1.4).

The example illustrates the main advantage offered by DP. While the original inventory problem requires an optimization over the set of policies, the DP algorithm of Eq. (1.4) decomposes this problem into a sequence of minimizations carried out over the set of controls. Each of these minimizations is much simpler than the original problem.

The DP Algorithm

We now state the DP algorithm for the basic problem and show its optimality by translating into mathematical terms the heuristic argument given above for the inventory example.

Proposition 1.3.1: For every initial state x_0, the optimal cost $J^*(x_0)$ of the basic problem is equal to $J_0(x_0)$, given by the last step of the following algorithm, which proceeds backward in time from period $N-1$ to period 0:

$$J_N(x_N) = g_N(x_N), \tag{1.5}$$

$$J_k(x_k) = \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\}, \qquad k = 0, 1, \ldots, N-1, \tag{1.6}$$

where the expectation is taken with respect to the probability distribution of w_k, which depends on x_k and u_k. Furthermore, if $u_k^* = \mu_k^*(x_k)$ minimizes the right side of Eq. (1.6) for each x_k and k, the policy $\pi^* = \{\mu_0^*, \ldots, \mu_{N-1}^*\}$ is optimal.

Proof:† For any admissible policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$ and each $k = 0, 1, \ldots, N-1$, denote $\pi^k = \{\mu_k, \mu_{k+1}, \ldots, \mu_{N-1}\}$. For $k = 0, 1, \ldots, N-1$, let $J_k^*(x_k)$ be the optimal cost for the $(N-k)$-stage problem that starts at state x_k and time k, and ends at time N,

$$J_k^*(x_k) = \min_{\pi^k} \mathop{E}_{w_k, \ldots, w_{N-1}} \left\{ g_N(x_N) + \sum_{i=k}^{N-1} g_i\big(x_i, \mu_i(x_i), w_i\big) \right\}.$$

For $k = N$, we define $J_N^*(x_N) = g_N(x_N)$. We will show by induction that the functions J_k^* are equal to the functions J_k generated by the DP algorithm, so that for $k = 0$, we will obtain the desired result.

Indeed, we have by definition $J_N^* = J_N = g_N$. Assume that for some k and all x_{k+1}, we have $J_{k+1}^*(x_{k+1}) = J_{k+1}(x_{k+1})$. Then, since $\pi^k = (\mu_k, \pi^{k+1})$, we have for all x_k

$$\begin{aligned} J_k^*(x_k) = \min_{(\mu_k, \pi^{k+1})} \mathop{E}_{w_k, \ldots, w_{N-1}} \Bigg\{ & g_k\big(x_k, \mu_k(x_k), w_k\big) \\ & + g_N(x_N) + \sum_{i=k+1}^{N-1} g_i\big(x_i, \mu_i(x_i), w_i\big) \Bigg\} \end{aligned}$$

† Our proof is somewhat informal and assumes that the functions J_k are well-defined and finite. For a strictly rigorous proof, some technical mathematical issues must be addressed; see Section 1.5. These issues do not arise if the disturbance w_k takes a finite or countable number of values and the expected values of all terms in the expression of the cost function (1.1) are well-defined and finite for every admissible policy π.

$$
\begin{aligned}
&= \min_{\mu_k} \mathop{E}_{w_k} \Bigg\{ g_k\big(x_k, \mu_k(x_k), w_k\big) \\
&\qquad + \min_{\pi^{k+1}} \left[\mathop{E}_{w_{k+1},\ldots,w_{N-1}} \left\{ g_N(x_N) + \sum_{i=k+1}^{N-1} g_i\big(x_i, \mu_i(x_i), w_i\big) \right\} \right] \Bigg\} \\
&= \min_{\mu_k} \mathop{E}_{w_k} \Big\{ g_k\big(x_k, \mu_k(x_k), w_k\big) + J^*_{k+1}\big(f_k\big(x_k, \mu_k(x_k), w_k\big)\big) \Big\} \\
&= \min_{\mu_k} \mathop{E}_{w_k} \Big\{ g_k\big(x_k, \mu_k(x_k), w_k\big) + J_{k+1}\big(f_k\big(x_k, \mu_k(x_k), w_k\big)\big) \Big\} \\
&= \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\} \\
&= J_k(x_k),
\end{aligned}
$$

completing the induction. In the second equation above, we moved the minimum over π^{k+1} inside the braced expression, using a principle of optimality argument: "the tail portion of an optimal policy is optimal for the tail subproblem" (a more rigorous justification of this step is given in Section 1.5). In the third equation, we used the definition of J^*_{k+1}, and in the fourth equation we used the induction hypothesis. In the fifth equation, we converted the minimization over μ_k to a minimization over u_k, using the fact that for any function F of x and u, we have

$$\min_{\mu \in M} F\big(x, \mu(x)\big) = \min_{u \in U(x)} F(x, u),$$

where M is the set of all functions $\mu(x)$ such that $\mu(x) \in U(x)$ for all x. **Q.E.D.**

The argument of the preceding proof provides an interpretation of $J_k(x_k)$ as the optimal cost for an $(N-k)$-stage problem starting at state x_k and time k, and ending at time N. We consequently call $J_k(x_k)$ the *cost-to-go* at state x_k and time k, and refer to J_k as the *cost-to-go function* or *optimal cost function* at time k.†

Ideally, we would like to use the DP algorithm to obtain closed-form expressions for J_k or an optimal policy. In this book, we will discuss quite a few models that admit analytical solution by DP. Even if such models rely on oversimplified assumptions, they are often very useful. They may provide valuable insights about the structure of the optimal solution of more complex models, and they may form the basis for suboptimal control schemes. Furthermore, the broad collection of analytically solvable models provides helpful guidelines for modeling: when faced with a new problem it

† In maximization problems the DP algorithm (1.6) is written with maximization in place of minimization, and then J_k is referred to as the *optimal value function* at time k.

is worth trying to pattern its model after one of the principal analytically tractable models.

Unfortunately, in many practical cases an analytical solution is not possible, and one has to resort to numerical execution of the DP algorithm. This may be quite time-consuming since the minimization in the DP Eq. (1.6) must be carried out for each value of x_k. The state space must be discretized in some way if it is not already a finite set. The computational requirements are proportional to the number of possible values of x_k, so for complex problems the computational burden may be excessive. Nonetheless, DP is the only general approach for sequential optimization under uncertainty, and even when it is computationally prohibitive, it can serve as the basis for more practical suboptimal approaches, which will be discussed in Chapter 6. Moreover, the DP computation is still far more economical than a brute force search, as the following example illustrates.

Example 1.3.1 (Complexity Aspects of DP)

Let us calculate more precisely the computational requirements of DP in a finite-spaces context. Assume that the state spaces $X_0, X_1, \ldots, X_{N-1}$ have no more than n elements each, and that at each state there are no more than m control elements available. Then the total number of state-control-time triples is no more than nmN. Thus nmN is an upper bound to the total number of times that expressions of the form

$$\underset{w_k}{E}\Big\{g_k(x_k,u_k,w_k) + J_{k+1}\big(f_k(x_k,u_k,w_k)\big)\Big\},$$

need to be calculated in the course of the DP algorithm [cf. Eq. (1.6)].

Of course the preceding expression may involve potentially significant computation. In particular, the expected value requires a number of calculations of the form

$$g_k(x_k,u_k,w_k) + J_{k+1}\big(f_k(x_k,u_k,w_k)\big). \tag{1.7}$$

This number is between 1 (for a deterministic problem) to n (typically, for a stochastic problem if the distribution of w_k is known). In either case, the number of times the expression (1.7) needs to be calculated is polynomial in n, m, and N.

Let us compare this computation with a brute force approach, which enumerates and compares all the possible solutions. A closed-loop policy $\{\mu_0, \ldots, \mu_{N-1}\}$ is characterized by a single control at each state-time pair (x_k, k), and there are no more than nN such pairs. With as many as m controls available at each of these pairs, we see that the number of distinct policies can be as many as m^{nN}. If we restrict ourselves to open-loop sequences (which we can for deterministic problems), still for a given initial condition, the number of possible sequences is as many as m^N. Thus the size of the solution space grows exponentially with N for both stochastic and deterministic problems.

A final observation is that the favorable complexity properties of DP depend critically on the additive structure of the cost function. We will see later in Section 1.4 that we can convert problems with a nonadditive cost structure to the basic problem format through a technique called *state augmentation*. However, in doing so the number of states grows again exponentially with N.

Dynamic Programming Examples

Let us now illustrate some of the analytical and computational aspects of DP by means of examples.

Example 1.3.2

We will go through the details of the DP algorithm for a stochastic inventory control problem that is similar to the one of Sections 1.1 and 1.2, but slightly different in some details. In particular, we assume that the inventory u_k and the demand w_k are nonnegative integers, and that the excess demand $(w_k - x_k - u_k)$ is lost. As a result, the stock equation takes the form

$$x_{k+1} = \max(0, x_k + u_k - w_k).$$

We also assume that there is an upper bound of 2 units on the stock that can be stored, i.e. there is a constraint $x_k + u_k \leq 2$. The holding/storage cost for the kth period is given by

$$(x_k + u_k - w_k)^2,$$

implying a penalty both for excess inventory and for unmet demand at the end of the kth period. The ordering cost is 1 per unit stock ordered. Thus the cost per period is

$$g_k(x_k, u_k, w_k) = u_k + (x_k + u_k - w_k)^2.$$

The terminal cost is assumed to be 0,

$$g_N(x_N) = 0.$$

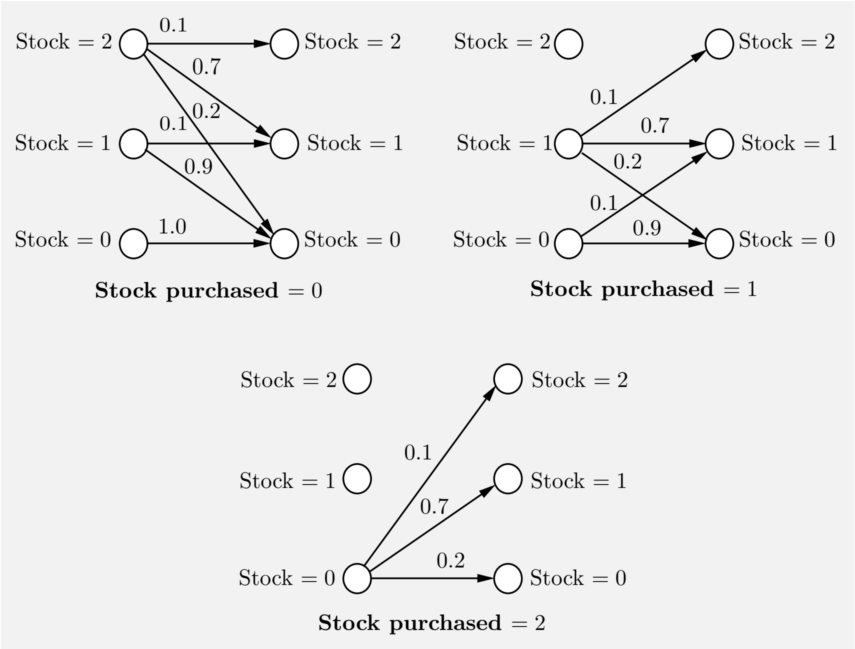
The planning horizon N is 3 periods, and the initial stock x_0 is 0. The demand w_k has the same probability distribution for all periods, given by

$$p(w_k = 0) = 0.1, \qquad p(w_k = 1) = 0.7, \qquad p(w_k = 2) = 0.2.$$

The system can also be represented in terms of the transition probabilities $p_{ij}(u)$ between the three possible states, for the different values of the control (see Fig. 1.3.2).

The starting equation for the DP algorithm is

$$J_3(x_3) = 0,$$



Stock	Stage 0 Cost-to-go	Stage 0 Optimal stock to purchase	Stage 1 Cost-to-go	Stage 1 Optimal stock to purchase	Stage 2 Cost-to-go	Stage 2 Optimal stock to purchase
0	3.7	1	2.5	1	1.3	1
1	2.7	0	1.5	0	0.3	0
2	2.818	0	1.68	0	1.1	0

Figure 1.3.2 System and DP results for Example 1.3.2. The transition probability diagrams for the different values of stock purchased (control) are shown. The numbers next to the arcs are the transition probabilities. The control $u = 1$ is not available at state 2 because of the limitation $x_k + u_k \leq 2$. Similarly, the control $u = 2$ is not available at states 1 and 2. The results of the DP algorithm are given in the table.

since the terminal state cost is 0 [cf. Eq. (1.5)]. The algorithm takes the form [cf. Eq. (1.6)]

$$J_k(x_k) = \min_{\substack{0 \leq u_k \leq 2 - x_k \\ u_k = 0,1,2}} \mathop{E}_{w_k} \Big\{ u_k + (x_k + u_k - w_k)^2 + J_{k+1}\big(\max(0, x_k + u_k - w_k)\big) \Big\},$$

where $k = 0, 1, 2,$ and x_k, u_k, w_k can take the values $0, 1,$ and 2.

Period 2: We compute $J_2(x_2)$ for each of the three possible states. We have

$$\begin{aligned} J_2(0) &= \min_{u_2=0,1,2} \mathop{E}_{w_2}\bigl\{u_2 + (u_2 - w_2)^2\bigr\} \\ &= \min_{u_2=0,1,2} \bigl[u_2 + 0.1(u_2)^2 + 0.7(u_2-1)^2 + 0.2(u_2-2)^2\bigr]. \end{aligned}$$

We calculate the expectation of the right side for each of the three possible values of u_2:

$$\begin{aligned} u_2 = 0 &: E\{\cdot\} = 0.7\cdot 1 + 0.2 \cdot 4 = 1.5, \\ u_2 = 1 &: E\{\cdot\} = 1 + 0.1\cdot 1 + 0.2\cdot 1 = 1.3, \\ u_2 = 2 &: E\{\cdot\} = 2 + 0.1\cdot 4 + 0.7\cdot 1 = 3.1. \end{aligned}$$

Hence we have, by selecting the minimizing u_2,

$$J_2(0) = 1.3, \qquad \mu_2^*(0) = 1.$$

For $x_2 = 1$, we have

$$\begin{aligned} J_2(1) &= \min_{u_2=0,1} \mathop{E}_{w_2}\bigl\{u_2 + (1 + u_2 - w_2)^2\bigr\} \\ &= \min_{u_2=0,1} \bigl[u_2 + 0.1(1+u_2)^2 + 0.7(u_2)^2 + 0.2(u_2-1)^2\bigr]. \end{aligned}$$

The expected value in the right side is

$$\begin{aligned} u_2 = 0 &: E\{\cdot\} = 0.1\cdot 1 + 0.2\cdot 1 = 0.3, \\ u_2 = 1 &: E\{\cdot\} = 1 + 0.1\cdot 4 + 0.7\cdot 1 = 2.1. \end{aligned}$$

Hence

$$J_2(1) = 0.3, \qquad \mu_2^*(1) = 0.$$

For $x_2 = 2$, the only admissible control is $u_2 = 0$, so we have

$$J_2(2) = \mathop{E}_{w_2}\bigl\{(2-w_2)^2\bigr\} = 0.1\cdot 4 + 0.7\cdot 1 = 1.1,$$

$$J_2(2) = 1.1, \qquad \mu_2^*(2) = 0.$$

Period 1: Again we compute $J_1(x_1)$ for each of the three possible states $x_1 = 0, 1, 2$, using the values $J_2(0)$, $J_2(1)$, $J_2(2)$ obtained in the previous period. For $x_1 = 0$, we have

$$J_1(0) = \min_{u_1=0,1,2} \mathop{E}_{w_1}\Bigl\{u_1 + (u_1 - w_1)^2 + J_2\bigl(\max(0, u_1 - w_1)\bigr)\Bigr\},$$

$$\begin{aligned} u_1 = 0 &: E\{\cdot\} = 0.1\cdot J_2(0) + 0.7\bigl(1 + J_2(0)\bigr) + 0.2\bigl(4 + J_2(0)\bigr) = 2.8, \\ u_1 = 1 &: E\{\cdot\} = 1 + 0.1\bigl(1 + J_2(1)\bigr) + 0.7\cdot J_2(0) + 0.2\bigl(1 + J_2(0)\bigr) = 2.5, \\ u_1 = 2 &: E\{\cdot\} = 2 + 0.1\bigl(4 + J_2(2)\bigr) + 0.7\bigl(1 + J_2(1)\bigr) + 0.2\cdot J_2(0) = 3.68, \end{aligned}$$

$$J_1(0) = 2.5, \qquad \mu_1^*(0) = 1.$$

For $x_1 = 1$, we have

$$J_1(1) = \min_{u_1=0,1} \mathop{E}_{w_1} \Big\{ u_1 + (1 + u_1 - w_1)^2 + J_2\big(\max(0, 1 + u_1 - w_1)\big) \Big\},$$

$$u_1 = 0 : E\{\cdot\} = 0.1\big(1 + J_2(1)\big) + 0.7 \cdot J_2(0) + 0.2\big(1 + J_2(0)\big) = 1.5,$$
$$u_1 = 1 : E\{\cdot\} = 1 + 0.1\big(4 + J_2(2)\big) + 0.7\big(1 + J_2(1)\big) + 0.2 \cdot J_2(0) = 2.68,$$

$$J_1(1) = 1.5, \qquad \mu_1^*(1) = 0.$$

For $x_1 = 2$, the only admissible control is $u_1 = 0$, so we have

$$\begin{aligned} J_1(2) &= \mathop{E}_{w_1} \Big\{ (2 - w_1)^2 + J_2\big(\max(0, 2 - w_1)\big) \Big\} \\ &= 0.1\big(4 + J_2(2)\big) + 0.7\big(1 + J_2(1)\big) + 0.2 \cdot J_2(0) \\ &= 1.68, \end{aligned}$$

$$J_1(2) = 1.68, \qquad \mu_1^*(2) = 0.$$

Period 0: Here we need to compute only $J_0(0)$ since the initial state is known to be 0. We have

$$J_0(0) = \min_{u_0=0,1,2} \mathop{E}_{w_0} \Big\{ u_0 + (u_0 - w_0)^2 + J_1\big(\max(0, u_0 - w_0)\big) \Big\},$$

$$u_0 = 0 : E\{\cdot\} = 0.1 \cdot J_1(0) + 0.7\big(1 + J_1(0)\big) + 0.2\big(4 + J_1(0)\big) = 4.0,$$
$$u_0 = 1 : E\{\cdot\} = 1 + 0.1\big(1 + J_1(1)\big) + 0.7 \cdot J_1(0) + 0.2\big(1 + J_1(0)\big) = 3.7,$$
$$u_0 = 2 : E\{\cdot\} = 2 + 0.1\big(4 + J_1(2)\big) + 0.7\big(1 + J_1(1)\big) + 0.2 \cdot J_1(0) = 4.818,$$

$$J_0(0) = 3.7, \qquad \mu_0^*(0) = 1.$$

If the initial state were not known a priori, we would have to compute in a similar manner $J_0(1)$ and $J_0(2)$, as well as the minimizing u_0. The reader may verify (Exercise 1.1) that these calculations yield

$$J_0(1) = 2.7, \qquad \mu_0^*(1) = 0,$$

$$J_0(2) = 2.818, \qquad \mu_0^*(2) = 0.$$

Thus the optimal ordering policy for each period is to order one unit if the current stock is zero and order nothing otherwise. The results of the DP algorithm are given in tabular form in Fig. 1.3.2.

Example 1.3.3 (A Linear-Quadratic Problem)

This is an example involving a one-dimensional linear system and a quadratic cost function. It illustrates an important class of problems that admit an analytical solution, and will be discussed in much greater detail later.

A certain material is passed through a sequence of two ovens (see Fig. 1.3.3). Denote

x_0: initial temperature of the material,

x_k, $k = 1, 2$: temperature of the material at the exit of oven k,

u_{k-1}, $k = 1, 2$: prevailing temperature in oven k.

We assume a model of the form

$$x_{k+1} = (1 - a)x_k + au_k, \qquad k = 0, 1,$$

where a is a known scalar from the interval $(0, 1)$. The objective is to get the final temperature x_2 close to a given target T, while expending relatively little energy. This is expressed by a cost function of the form

$$r(x_2 - T)^2 + u_0^2 + u_1^2,$$

where $r > 0$ is a given scalar. We assume no constraints on u_k. (In reality, there are constraints, but if we can solve the unconstrained problem and verify that the solution satisfies the constraints, everything will be fine.) The problem is deterministic; i.e., there is no stochastic uncertainty. However, such problems can be placed within the basic framework by introducing a fictitious disturbance taking a unique value with probability one.

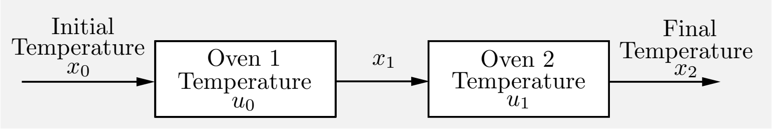


Figure 1.3.3 The linear-quadratic problem of Example 1.3.3. The temperature of the material evolves according to $x_{k+1} = (1 - a)x_k + au_k$, where a is some scalar with $0 < a < 1$.

We have $N = 2$ and a terminal cost $g_2(x_2) = r(x_2 - T)^2$, so the initial condition for the DP algorithm is [cf. Eq. (1.5)]

$$J_2(x_2) = r(x_2 - T)^2.$$

For the next-to-last stage, we have [cf. Eq. (1.6)]

$$\begin{aligned} J_1(x_1) &= \min_{u_1} \big[u_1^2 + J_2(x_2)\big] \\ &= \min_{u_1} \Big[u_1^2 + J_2\big((1 - a)x_1 + au_1\big)\Big]. \end{aligned}$$

Substituting the previous form of J_2, we obtain

$$J_1(x_1) = \min_{u_1} \left[u_1^2 + r\big((1-a)x_1 + au_1 - T\big)^2 \right]. \tag{1.8}$$

This minimization will be done by setting to zero the derivative with respect to u_1. This yields

$$0 = 2u_1 + 2ra\big((1-a)x_1 + au_1 - T\big),$$

and by collecting terms and solving for u_1, we obtain the optimal temperature for the last oven:

$$\mu_1^*(x_1) = \frac{ra\big(T - (1-a)x_1\big)}{1+ra^2}.$$

Note that this is not a single control but rather a control function, a rule that tells us the optimal oven temperature $u_1 = \mu_1^*(x_1)$ for each possible value of the state x_1.

By substituting the optimal u_1 in the expression (1.8) for J_1, we obtain

$$\begin{aligned} J_1(x_1) &= \frac{r^2a^2\big((1-a)x_1 - T\big)^2}{(1+ra^2)^2} + r\left((1-a)x_1 + \frac{ra^2\big(T-(1-a)x_1\big)}{1+ra^2} - T\right)^2 \\ &= \frac{r^2a^2\big((1-a)x_1 - T\big)^2}{(1+ra^2)^2} + r\left(\frac{ra^2}{1+ra^2} - 1\right)^2 \big((1-a)x_1 - T\big)^2 \\ &= \frac{r\big((1-a)x_1 - T\big)^2}{1+ra^2}. \end{aligned}$$

We now go back one stage. We have [cf. Eq. (1.6)]

$$J_0(x_0) = \min_{u_0} \big[u_0^2 + J_1(x_1)\big] = \min_{u_0} \left[u_0^2 + J_1\big((1-a)x_0 + au_0\big)\right],$$

and by substituting the expression already obtained for J_1, we have

$$J_0(x_0) = \min_{u_0} \left[u_0^2 + \frac{r\big((1-a)^2x_0 + (1-a)au_0 - T\big)^2}{1+ra^2} \right].$$

We minimize with respect to u_0 by setting the corresponding derivative to zero. We obtain

$$0 = 2u_0 + \frac{2r(1-a)a\big((1-a)^2x_0 + (1-a)au_0 - T\big)}{1+ra^2}.$$

This yields, after some calculation, the optimal temperature of the first oven:

$$\mu_0^*(x_0) = \frac{r(1-a)a\big(T - (1-a)^2x_0\big)}{1 + ra^2\big(1 + (1-a)^2\big)}.$$

The optimal cost is obtained by substituting this expression in the formula for J_0. This leads to a straightforward but lengthy calculation, which in the end yields the rather simple formula

$$J_0(x_0) = \frac{r\big((1-a)^2 x_0 - T\big)^2}{1 + ra^2\big(1 + (1-a)^2\big)}.$$

This completes the solution of the problem.

One noteworthy feature in this example is the facility with which we obtained an analytical solution. A little thought while tracing the steps of the algorithm will convince the reader that what simplifies the solution is the quadratic nature of the cost and the linearity of the system equation. In Section 3.1 we will see that, generally, when the system is linear and the cost is quadratic, the optimal policy and cost-to-go function are given by closed-form expressions, regardless of the number of stages N.

Another noteworthy feature of the example is that the optimal policy remains unaffected when a zero-mean stochastic disturbance is added in the system equation. To see this, assume that the material's temperature evolves according to

$$x_{k+1} = (1-a)x_k + au_k + w_k, \qquad k = 0, 1,$$

where w_0, w_1 are independent random variables with given distribution, zero mean

$$E\{w_0\} = E\{w_1\} = 0,$$

and finite variance. Then the equation for J_1 [cf. Eq. (1.6)] becomes

$$\begin{aligned} J_1(x_1) &= \min_{u_1} \mathop{E}_{w_1} \left\{ u_1^2 + r\big((1-a)x_1 + au_1 + w_1 - T\big)^2 \right\} \\ &= \min_{u_1} \Big[u_1^2 + r\big((1-a)x_1 + au_1 - T\big)^2 \\ &\qquad + 2rE\{w_1\}\big((1-a)x_1 + au_1 - T\big) + rE\{w_1^2\} \Big]. \end{aligned}$$

Since $E\{w_1\} = 0$, we obtain

$$J_1(x_1) = \min_{u_1} \Big[u_1^2 + r\big((1-a)x_1 + au_1 - T\big)^2 \Big] + rE\{w_1^2\}.$$

Comparing this equation with Eq. (1.8), we see that the presence of w_1 has resulted in an additional inconsequential constant term, $rE\{w_1^2\}$. Therefore, the optimal policy for the last stage remains unaffected by the presence of w_1, while $J_1(x_1)$ is increased by $rE\{w_1^2\}$. It can be seen that a similar situation also holds for the first stage. In particular, the optimal cost is given by the same expression as before except for an additive constant that depends on $E\{w_0^2\}$ and $E\{w_1^2\}$.

If the optimal policy is unaffected when the disturbances are replaced by their means, we say that *certainty equivalence* holds. We will derive certainty equivalence results for several types of problems involving a linear system and a quadratic cost (see Sections 3.1, 4.2, and 4.3).

Example 1.3.4 (Optimizing a Chess Match Strategy)

Consider the chess match Example 1.1.5. There, a player can select timid play (probabilities p_d and $1-p_d$ for a draw or loss, respectively) or bold play (probabilities p_w and $1-p_w$ for a win or loss, respectively) in each game of the match. We want to formulate a DP algorithm for finding the policy that maximizes the player's probability of winning the match. Note that here we are dealing with a maximization problem. We can convert the problem to a minimization problem by changing the sign of the cost function, but a simpler alternative, which we will generally adopt, is to replace the minimization in the DP algorithm with maximization.

Let us consider the general case of an N-game match, and let the state be the *net score*, i.e., the difference between the points of the player minus the points of the opponent (so a state of 0 corresponds to an even score). The optimal cost-to-go function at the start of the kth game is given by the DP recursion

$$\begin{aligned} J_k(x_k) = \max\big[&p_d J_{k+1}(x_k) + (1-p_d)J_{k+1}(x_k-1), \\ &p_w J_{k+1}(x_k+1) + (1-p_w)J_{k+1}(x_k-1)\big]. \end{aligned} \tag{1.9}$$

The maximum above is taken over the two possible decisions:

(a) Timid play, which keeps the score at x_k with probability p_d, and changes x_k to $x_k - 1$ with probability $1-p_d$.

(b) Bold play, which changes x_k to x_k+1 or to x_k-1 with probabilities p_w or $(1-p_w)$, respectively.

It is optimal to play bold when

$$p_w J_{k+1}(x_k+1) + (1-p_w)J_{k+1}(x_k-1) \geq p_d J_{k+1}(x_k) + (1-p_d)J_{k+1}(x_k-1)$$

or equivalently, if

$$\frac{p_w}{p_d} \geq \frac{J_{k+1}(x_k) - J_{k+1}(x_k-1)}{J_{k+1}(x_k+1) - J_{k+1}(x_k-1)}. \tag{1.10}$$

The DP recursion is started with

$$J_N(x_N) = \begin{cases} 1 & \text{if } x_N > 0, \\ p_w & \text{if } x_N = 0, \\ 0 & \text{if } x_N < 0. \end{cases} \tag{1.11}$$

In this equation, we have $J_N(0) = p_w$ because when the score is even after N games ($x_N = 0$), it is optimal to play bold in the first game of sudden death.

By executing the DP algorithm (1.9) starting with the terminal condition (1.11), and using the criterion (1.10) for optimality of bold play, we find the following, assuming that $p_d > p_w$:

$$\begin{aligned} J_{N-1}(x_{N-1}) &= 1 \text{ for } x_{N-1} > 1; \quad \text{optimal play: either} \\ J_{N-1}(1) &= \max[p_d + (1-p_d)p_w,\, p_w + (1-p_w)p_w] \\ &= p_d + (1-p_d)p_w; \quad \text{optimal play: timid} \\ J_{N-1}(0) &= p_w; \quad \text{optimal play: bold} \\ J_{N-1}(-1) &= p_w^2; \quad \text{optimal play: bold} \\ J_{N-1}(x_{N-1}) &= 0 \text{ for } x_{N-1} < -1; \quad \text{optimal play: either.} \end{aligned}$$

Also, given $J_{N-1}(x_{N-1})$, and using Eqs. (1.9) and (1.10), we obtain

$$\begin{aligned} J_{N-2}(0) &= \max\Big[p_d p_w + (1-p_d)p_w^2,\ p_w\big(p_d + (1-p_d)p_w\big) + (1-p_w)p_w^2\Big] \\ &= p_w\big(p_w + (p_w+p_d)(1-p_w)\big), \end{aligned}$$

and that if the score is even with 2 games remaining, it is optimal to play bold. Thus for a 2-game match, the optimal policy for both periods is to play timid if and only if the player is ahead in the score. The region of pairs (p_w, p_d) for which the player has a better than 50-50 chance to win a 2-game match is

$$R_2 = \Big\{(p_w,p_d) \mid J_0(0) = p_w\big(p_w + (p_w+p_d)(1-p_w)\big) > 1/2\Big\},$$

and, as noted in Example 1.2.1, it includes points where $p_w < 1/2$.

Example 1.3.5 (Finite State Systems)

We mentioned earlier (cf. the examples in Section 1.1) that systems with a finite number of states can be represented either in terms of a discrete-time system equation or in terms of the probabilities of transition between the states. Let us work out the DP algorithm corresponding to the latter case. We assume for the sake of the following discussion that the problem is stationary, i.e., the transition probabilities, the cost per stage, and the control constraint sets do not change from one stage to the next. Then, if

$$p_{ij}(u) = P\{x_{k+1} = j \mid x_k = i, u_k = u\}$$

are the transition probabilities, we can alternatively represent the system by the system equation (cf. the discussion of the previous section)

$$x_{k+1} = w_k,$$

where the probability distribution of the disturbance w_k is

$$P\{w_k = j \mid x_k = i, u_k = u\} = p_{ij}(u).$$

Using this system equation and denoting by $g(i,u)$ the expected cost per stage at state i when control u is applied, the DP algorithm can be rewritten as

$$J_k(i) = \min_{u\in U(i)} \Big[g(i,u) + E\big\{J_{k+1}(w_k)\big\}\Big]$$

or equivalently (in view of the distribution of w_k given previously)

$$J_k(i) = \min_{u\in U(i)} \left[g(i,u) + \sum_j p_{ij}(u) J_{k+1}(j)\right].$$

As an illustration, in the machine replacement Example 1.1.3, this algorithm takes the form

$$J_N(i) = 0, \qquad i = 1, \ldots, n,$$

$$J_k(i) = \min \left[R + g(1) + J_{k+1}(1),\, g(i) + \sum_{j=i}^{n} p_{ij} J_{k+1}(j) \right].$$

The two expressions in the above minimization correspond to the two available decisions (replace or not replace the machine).

In the queueing Example 1.1.4, the DP algorithm takes the form

$$J_N(i) = R(i), \qquad i = 0, 1, \ldots, n,$$

$$J_k(i) = \min \left[r(i) + c_f + \sum_{j=0}^{n} p_{ij}(u_f) J_{k+1}(j),\, r(i) + c_s + \sum_{j=0}^{n} p_{ij}(u_s) J_{k+1}(j) \right].$$

The two expressions in the above minimization correspond to the two possible decisions (fast and slow service).

1.4 STATE AUGMENTATION AND OTHER REFORMULATIONS

We now discuss how to deal with situations where some of the assumptions of the basic problem are violated. Generally, in such cases the problem can be reformulated into the basic problem format. This process is called *state augmentation* because it typically involves the enlargement of the state space. The general guideline in state augmentation is to *include in the enlarged state at time k all the information that is known to the controller at time k and can be used with advantage in selecting u_k*. Unfortunately, state augmentation often comes at a price: the reformulated problem may have very complex state and/or control spaces. We provide some examples.

Time Delays

In many applications the system state x_{k+1} depends not only on the preceding state x_k and control u_k but also on earlier states and controls. In other words, states and controls influence future states with some time delay. Such situations can be handled by state augmentation; the state is expanded to include an appropriate number of earlier states and controls.

For simplicity, assume that there is at most a single period time delay in the state and control; i.e., the system equation has the form

$$x_{k+1} = f_k(x_k, x_{k-1}, u_k, u_{k-1}, w_k), \qquad k = 1, 2, \ldots, N-1, \tag{1.12}$$

$$x_1 = f_0(x_0, u_0, w_0).$$

Time delays of more than one period can be handled similarly.

If we introduce additional state variables y_k and s_k, and we make the identifications $y_k = x_{k-1}$, $s_k = u_{k-1}$, the system equation (1.12) yields

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \\ s_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, y_k, u_k, s_k, w_k) \\ x_k \\ u_k \end{pmatrix}. \tag{1.13}$$

By defining $\tilde{x}_k = (x_k, y_k, s_k)$ as the new state, we have

$$\tilde{x}_{k+1} = \tilde{f}_k(\tilde{x}_k, u_k, w_k),$$

where the system function $\tilde{f}_k$ is defined from Eq. (1.13). By using the preceding equation as the system equation and by expressing the cost function in terms of the new state, the problem is reduced to the basic problem without time delays. Naturally, the control u_k should now depend on the new state $\tilde{x}_k$, or equivalently a policy should consist of functions μ_k of the current state x_k, as well as the preceding state x_{k-1} and the preceding control u_{k-1}.

When the DP algorithm for the reformulated problem is translated in terms of the variables of the original problem, it takes the form

$$J_N(x_N) = g_N(x_N),$$

$$\begin{aligned} &J_{N-1}(x_{N-1}, x_{N-2}, u_{N-2}) \\ &\quad = \min_{u_{N-1} \in U_{N-1}(x_{N-1})} \mathop{E}_{w_{N-1}} \Big\{ g_{N-1}(x_{N-1}, u_{N-1}, w_{N-1}) \\ &\qquad\qquad + J_N\big(f_{N-1}(x_{N-1}, x_{N-2}, u_{N-1}, u_{N-2}, w_{N-1})\big) \Big\}, \end{aligned}$$

$$\begin{aligned} J_k(x_k, x_{k-1}, u_{k-1}) &= \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ g_k(x_k, u_k, w_k) \\ &+ J_{k+1}\big(f_k(x_k, x_{k-1}, u_k, u_{k-1}, w_k), x_k, u_k\big) \Big\}, \quad k = 1, \ldots, N-2, \end{aligned}$$

$$J_0(x_0) = \min_{u_0 \in U_0(x_0)} \mathop{E}_{w_0} \Big\{ g_0(x_0, u_0, w_0) + J_1\big(f_0(x_0, u_0, w_0), x_0, u_0\big) \Big\}.$$

Similar reformulations are possible when time delays appear in the cost; for example, in the case where the cost has the form

$$E\left\{ g_N(x_N, x_{N-1}) + g_0(x_0, u_0, w_0) + \sum_{k=1}^{N-1} g_k(x_k, x_{k-1}, u_k, w_k) \right\}.$$

The extreme case of time delays in the cost arises in the nonadditive form

$$E\big\{ g_N(x_N, x_{N-1}, \ldots, x_0, u_{N-1}, \ldots, u_0, w_{N-1}, \ldots, w_0) \big\}.$$

Then, the problem can be reduced to the basic problem format, by taking as augmented state

$$\tilde{x}_k = \big(x_k, x_{k-1}, \ldots, x_0, u_{k-1}, \ldots, u_0, w_{k-1}, \ldots, w_0\big)$$

and $E\big\{g_N(\tilde{x}_N)\big\}$ as reformulated cost. Policies consist of functions μ_k of the present and past states $x_k, \ldots, x_0$, the past controls $u_{k-1}, \ldots, u_0$, and the past disturbances $w_{k-1}, \ldots, w_0$. Naturally, we must assume that the past disturbances are known to the controller. Otherwise, we are faced with a problem where the state is imprecisely known to the controller. Such problems are known as problems with imperfect state information and will be discussed in Chapter 4.

Correlated Disturbances

Consider the case where the disturbances w_k are correlated over time. A common situation that can be handled efficiently by state augmentation arises when the process $w_0, \ldots, w_{N-1}$ can be represented as the output of a linear system driven by independent random variables. As an example, suppose that by using statistical methods, we determine that the evolution of w_k can be modeled by an equation of the form

$$w_k = \lambda w_{k-1} + \xi_k,$$

where λ is a given scalar and $\{\xi_k\}$ is a sequence of independent random vectors with given distribution. Then we can introduce an additional state variable

$$y_k = w_{k-1}$$

and obtain a new system equation

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, u_k, \lambda y_k + \xi_k) \\ \lambda y_k + \xi_k \end{pmatrix},$$

where the new state is the pair $\tilde{x}_k = (x_k, y_k)$ and the new disturbance is the vector ξ_k.

More generally, suppose that w_k can be modeled by

$$w_k = C_k y_{k+1},$$

where

$$y_{k+1} = A_k y_k + \xi_k, \qquad k = 0, \ldots, N-1,$$

A_k, C_k are known matrices of appropriate dimension, and ξ_k are independent random vectors with given distribution (see Fig. 1.4.1). By viewing y_k as an additional state variable, we obtain the new system equation

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k\big(x_k, u_k, C_k(A_k y_k + \xi_k)\big) \\ A_k y_k + \xi_k \end{pmatrix}.$$

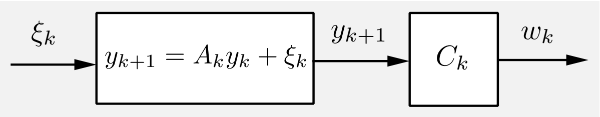


Figure 1.4.1 Representing correlated disturbances as the output of a linear system driven by independent random vectors.

Note that in order to have perfect state information, the controller must be able to observe y_k. Unfortunately, this is true only in the minority of practical cases; for example when C_k is the identity matrix and w_{k-1} is observed before u_k is applied. In the case of perfect state information, the DP algorithm takes the form

$$J_N(x_N, y_N) = g_N(x_N),$$

$$\begin{aligned} J_k(x_k, y_k) = \min_{u_k \in U_k(x_k)} \mathop{E}_{\xi_k} \Big\{ & g_k\big(x_k, u_k, C_k(A_k y_k + \xi_k)\big) \\ & + J_{k+1}\Big(f_k\big(x_k, u_k, C_k(A_k y_k + \xi_k)\big), A_k y_k + \xi_k\Big)\Big\}. \end{aligned}$$

Forecasts

Consider the case where at time k the controller has access to a forecast y_k that results in a reassessment of the probability distribution of w_k and possibly of future disturbances. For example, y_k may be an exact prediction of w_k or an exact prediction that the probability distribution of w_k is a specific one out of a finite collection of distributions. Forecasts of interest in practice are, for example, probabilistic predictions on the state of the weather, the interest rate for money, and the demand for inventory.

Generally, forecasts can be handled by state augmentation although the reformulation into the basic problem format may be quite complex. We will treat here only a simple special case.

Assume that at the beginning of each period k, the controller receives an accurate prediction that the next disturbance w_k will be selected according to a particular probability distribution out of a given collection of distributions $\{Q_1, \ldots, Q_m\}$; i.e., if the forecast is i, then w_k is selected according to Q_i. The a priori probability that the forecast will be i is denoted by p_i and is given.

For instance, suppose that in our earlier inventory example the demand w_k is determined according to one of three distributions Q_1, Q_2, and Q_3, corresponding to "small," "medium," and "large" demand. Each of the three types of demand occurs with a given probability at each time period, independently of the values of demand at previous time periods. However, the inventory manager, prior to ordering u_k, gets to know through a forecast the type of demand that will occur. (Note that it is the probability distribution of demand that becomes known through the forecast, not the demand itself.)

The forecasting process can be represented by means of the equation

$$y_{k+1} = \xi_k,$$

where y_{k+1} can take the values $1, \ldots, m$, corresponding to the m possible forecasts, and ξ_k is a random variable taking the value i with probability p_i. The interpretation here is that when ξ_k takes the value i, then w_{k+1} will occur according to the distribution Q_i.

By combining the system equation with the forecast equation $y_{k+1} = \xi_k$, we obtain an augmented system given by

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, u_k, w_k) \\ \xi_k \end{pmatrix}.$$

The new state is

$$\tilde{x}_k = (x_k, y_k),$$

and because the forecast y_k is known at time k, perfect state information prevails. The new disturbance is

$$\tilde{w}_k = (w_k, \xi_k),$$

and its probability distribution is determined by the distributions Q_i and the probabilities p_i, and depends explicitly on $\tilde{x}_k$ (via y_k) but not on the prior disturbances.

Thus, by suitable reformulation of the cost, the problem can be cast into the basic problem format. Note that the control applied depends on both the current state and the current forecast. The DP algorithm takes the form

$$J_N(x_N, y_N) = g_N(x_N),$$

$$\begin{aligned} J_k(x_k, y_k) = \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ & g_k(x_k, u_k, w_k) \\ & + \sum_{i=1}^{m} p_i J_{k+1}\big(f_k(x_k, u_k, w_k), i\big) \mid y_k \Big\}, \end{aligned} \qquad (1.14)$$

where y_k may take the values $1, \ldots, m$, and the expectation over w_k is taken with respect to the distribution Q_{y_k}.

It should be clear that the preceding formulation admits several extensions. One example is the case where forecasts can be influenced by the control action and involve several future disturbances. However, the price for these extensions is increased complexity of the corresponding DP algorithm.

Simplification for Uncontrollable State Components

When augmenting the state of a given system we often end up with composite states, consisting of several components. It turns out that if some of these components cannot be affected by the choice of control, the DP algorithm can be simplified considerably, as we will now describe.

Let the state of the system be a composite (x_k, y_k) of two components x_k and y_k. The evolution of the main component, x_k, is affected by the control u_k according to the equation

$$x_{k+1} = f_k(x_k, y_k, u_k, w_k),$$

where the probability distribution $P_k(w_k \mid x_k, y_k, u_k)$ is given. The evolution of the other component, y_k, is governed by a given conditional distribution $P_k(y_k \mid x_k)$ and cannot be affected by the control, except indirectly through x_k. One is tempted to view y_k as a disturbance, but there is a difference: y_k is observed by the controller before applying u_k, while w_k occurs after u_k is applied, and indeed w_k may probabilistically depend on u_k.

We will formulate a DP algorithm that is executed over the controllable component of the state, with the dependence on the uncontrollable component being "averaged out." In particular, let $J_k(x_k, y_k)$ denote the optimal cost-to-go at stage k and state (x_k, y_k), and define

$$\hat{J}_k(x_k) = \mathop{E}_{y_k}\big\{J_k(x_k, y_k) \mid x_k\big\}.$$

We will derive a DP algorithm that generates $\hat{J}_k(x_k)$.

Indeed, we have

$$\begin{aligned}
\hat{J}_k(x_k) &= E_{y_k}\big\{J_k(x_k, y_k) \mid x_k\big\} \\
&= E_{y_k}\Big\{\min_{u_k \in U_k(x_k, y_k)} E_{w_k, x_{k+1}, y_{k+1}}\big\{g_k(x_k, y_k, u_k, w_k) \\
&\qquad + J_{k+1}(x_{k+1}, y_{k+1}) \bigm| x_k, y_k, u_k\big\} \bigm| x_k\Big\} \\
&= E_{y_k}\Big\{\min_{u_k \in U_k(x_k, y_k)} E_{w_k, x_{k+1}}\Big\{g_k(x_k, y_k, u_k, w_k) \\
&\qquad + E_{y_{k+1}}\big\{J_{k+1}(x_{k+1}, y_{k+1}) \bigm| x_{k+1}\big\} \bigm| x_k, y_k, u_k\Big\} \bigm| x_k\Big\},
\end{aligned}$$

and finally

$$\begin{aligned}
\hat{J}_k(x_k) = \mathop{E}_{y_k}\Big\{ \min_{u_k \in U_k(x_k, y_k)} \mathop{E}_{w_k}\Big\{g_k(x_k, y_k, u_k, w_k) \\
+ \hat{J}_{k+1}\big(f_k(x_k, y_k, u_k, w_k)\big)\Big\} \Bigm| x_k\Big\}.
\end{aligned} \tag{1.15}$$

The advantage of this equivalent DP algorithm is that it is executed over a significantly reduced state space. For example, if x_k takes n possible values and y_k takes m possible values, then DP is executed over n states instead of nm states. Note, however, that the minimization in the right-hand side of the preceding equation yields an optimal control law as a function of the full state (x_k, y_k).

As an example, consider the augmented state resulting from the incorporation of forecasts, as described earlier in this section. Then, the forecast y_k represents an uncontrolled state component, so that the DP algorithm can be simplified as in Eq. (1.15). In particular, by defining

$$\hat{J}_k(x_k) = \sum_{i=1}^{m} p_i J_k(x_k, i), \qquad k = 0, 1, \ldots, N-1,$$

and

$$\hat{J}_N(x_N) = g_N(x_N),$$

we have, using Eq. (1.14),

$$\hat{J}_k(x_k) = \sum_{i=1}^{m} p_i \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ g_k(x_k, u_k, w_k) + \hat{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \mid y_k = i \Big\},$$

which is executed over the space of x_k rather than x_k *and* y_k. This is a simpler algorithm than the one of Eq. (1.14).

Uncontrollable state components often occur in arrival systems, such as queueing, where action must be taken in response to a random event (such as a customer arrival) that cannot be influenced by the choice of control. Then the state of the arrival system must be augmented to include the random event, but the DP algorithm can be executed over a smaller space, as per Eq. (1.15). Here is another example of similar type.

Example 1.4.1 (Tetris)

Tetris is a popular video game played on a two-dimensional grid. Each square in the grid can be full or empty, making up a "wall of bricks" with "holes" and a "jagged top." The squares fill up as blocks of different shapes fall from the top of the grid and are added to the top of the wall. As a given block falls, the player can move horizontally and rotate the block in all possible ways, subject to the constraints imposed by the sides of the grid and the top of the wall. The falling blocks are generated independently according to some probability distribution, defined over a finite set of standard shapes. The game starts with an empty grid and ends when a square in the top row becomes full and the top of the wall reaches the top of the grid. When a row of full squares is created, this row is removed, the bricks lying above this row move one row downward, and the player scores a point. The player's objective is to maximize the score attained (total number of rows removed) within N steps or up to termination of the game, whichever occurs first.

We can model the problem of finding an optimal tetris playing strategy as a stochastic DP problem. The control, denoted by u, is the horizontal positioning and rotation applied to the falling block. The state consists of two components:

(1) The board position, i.e., a binary description of the full/empty status of each square, denoted by x.

(2) The shape of the current falling block, denoted by y.

There is also an additional termination state which is cost-free. Once the state reaches the termination state, it stays there with no change in cost.

The shape y is generated according to a probability distribution $p(y)$, independently of the control, so it can be viewed as an uncontrollable state component. The DP algorithm (1.15) is executed over the space of x and has the intuitive form

$$\hat{J}_k(x) = \sum_y p(y) \max_u \Big[g(x,y,u) + \hat{J}_{k+1}\big(f(x,y,u)\big)\Big], \qquad \text{for all } x,$$

where $g(x,y,u)$ and $f(x,y,u)$ are the number of points scored (rows removed), and the board position (or termination state) when the state is (x,y) and control u is applied, respectively. Note, however, that despite the simplification in the DP algorithm achieved by eliminating the uncontrollable portion of the state, the number of states x is enormous, and the problem can only be addressed by suboptimal methods, which will be discussed in Chapter 6 and in Vol. II.

1.5 SOME MATHEMATICAL ISSUES

Let us now discuss some technical issues relating to the basic problem formulation and the validity of the DP algorithm. The reader who is not mathematically inclined need not be concerned about these issues and can skip this section without loss of continuity; the mathematical fine points do not contribute significantly to the intuition for solving practical problems and do not matter if the disturbances w_k can take only a finite number of values.

Once an admissible policy $\{\mu_0, \ldots, \mu_{N-1}\}$ is adopted, the following sequence of events is envisioned at the typical stage k:

1. The controller observes x_k and applies $u_k = \mu_k(x_k)$.
2. The disturbance w_k is generated according to the given distribution $P_k\big(\cdot \mid x_k, \mu_k(x_k)\big)$.
3. The cost $g_k\big(x_k, \mu_k(x_k), w_k\big)$ is incurred and added to previous costs.

4. The next state x_{k+1} is generated according to the system equation

$$x_{k+1} = f_k\bigl(x_k, \mu_k(x_k), w_k\bigr).$$

If this is the last stage ($k = N - 1$), the terminal cost $g_N(x_N)$ is added to previous costs and the process terminates. Otherwise, k is incremented, and the same sequence of events is repeated at the next stage.

For each stage, the above process is well-defined and is couched in precise probabilistic terms. Matters are, however, complicated by the need to view the cost as a well-defined random variable with well-defined expected value. The framework of probability theory requires that for each policy we define an underlying probability space, i.e., a set Ω, a collection of events in Ω, and a probability measure on these events. In addition, the cost must be a well-defined random variable on this space in the sense of Appendix C (a measurable function from the probability space into the real line in the terminology of measure-theoretic probability theory). For this to be true, additional (measurability) assumptions on the functions f_k, g_k, and μ_k may be required, and it may be necessary to introduce additional structure on the spaces S_k, C_k, and D_k. Furthermore, these assumptions may restrict the class of admissible policies, since the functions μ_k may be constrained to satisfy additional (measurability) requirements.

Thus, unless these additional assumptions and structure are specified, the basic problem is formulated inadequately from a mathematical point of view. Unfortunately, a rigorous formulation for general state, control, and disturbance spaces is well beyond the mathematical framework of this introductory book and will not be undertaken here. Nonetheless, it turns out that these difficulties are mainly technical and do not substantially affect the basic results to be obtained. For this reason, we find it convenient to proceed with informal derivations and arguments; this is consistent with most of the literature on the subject.

We would like to stress, however, that under at least one frequently satisfied assumption, the mathematical difficulties mentioned above disappear. In particular, let us assume that the disturbance spaces D_k are all countable and the expected values of all terms in the cost are finite for every admissible policy (this is true in particular if the spaces D_k are finite sets). Then, for every admissible policy, the expected values of all the cost terms can be written as (possibly infinite) sums involving the probabilities of the elements of the spaces D_k, and no measurability framework is needed.

Alternatively, one may write the cost as

$$J_\pi(x_0) = \mathop{E}_{x_1,\ldots,x_N}\left\{ g_N(x_N) + \sum_{k=0}^{N-1} \tilde{g}_k\bigl(x_k, \mu_k(x_k)\bigr) \right\}, \tag{1.16}$$

where

$$\tilde{g}_k\bigl(x_k, \mu_k(x_k)\bigr) = \mathop{E}_{w_k}\Bigl\{ g_k\bigl(x_k, \mu_k(x_k), w_k\bigr) \mid x_k, \mu_k(x_k) \Bigr\},$$

with the preceding expectation taken with respect to the distribution $P_k\big(\cdot \mid x_k, \mu_k(x_k)\big)$ defined on the countable set D_k. Then one may take as the basic probability space the Cartesian product of the spaces $\tilde{S}_k$, $k = 1, \ldots, N$, given for all k by

$$\tilde{S}_{k+1} = \big\{x_{k+1} \in S_{k+1} \mid x_{k+1} = f_k\big(x_k, \mu_k(x_k), w_k\big),\ x_k \in \tilde{S}_k,\ w_k \in D_k\big\},$$

where $\tilde{S}_0 = \{x_0\}$. The set $\tilde{S}_k$ is the subset of all states that can be reached at time k when the policy $\{\mu_0, \ldots, \mu_{N-1}\}$ is used. Because the disturbance spaces D_k are countable, the sets $\tilde{S}_k$ are also countable (this is true since the union of any countable collection of countable sets is a countable set). The system equation $x_{k+1} = f_k\big(x_k, \mu_k(x_k), w_k\big)$, the probability distributions $P_k\big(\cdot \mid x_k, \mu_k(x_k)\big)$, the initial state x_0, and the policy $\{\mu_0, \ldots, \mu_{N-1}\}$ define a probability distribution on the countable set $\tilde{S}_1 \times \cdots \times \tilde{S}_N$, and the expected value in the cost expression (1.16) is defined with respect to this latter distribution.

Let us now give a more detailed proof of the validity of the DP algorithm (Prop. 1.3.1). We assume that the disturbance w_k takes a finite or countable number of values and the expected values of all terms in the expression of the cost function are finite for every admissible policy π. Furthermore, the functions $J_k(x_k)$ generated by the DP algorithm are finite for all states x_k and times k. We do not need to assume that the minimum over u_k in the definition of $J_k(x_k)$ is attained by some $u_k \in U(x_k)$.

For any admissible policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$ and each $k = 0, 1, \ldots, N-1$, denote $\pi^k = \{\mu_k, \mu_{k+1}, \ldots, \mu_{N-1}\}$. For $k = 0, 1, \ldots, N-1$, let $J_k^*(x_k)$ be the optimal cost for the $(N-k)$-stage problem that starts at state x_k and time k, and ends at time N; i.e.,

$$J_k^*(x_k) = \min_{\pi^k} E\left\{ g_N(x_N) + \sum_{i=k}^{N-1} g_i\big(x_i, \mu_i(x_i), w_i\big) \right\}.$$

For $k = N$, we define $J_N^*(x_N) = g_N(x_N)$. We will show by induction that the functions J_k^* are equal to the functions J_k generated by the DP algorithm, so that for $k = 0$, we will obtain the desired result.

For any $\epsilon > 0$, and for all k and x_k, let $\mu_k^\epsilon(x_k)$ attain the minimum in the equation

$$J_k(x_k) = \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k}\big\{g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, u_k, w_k)\big)\big\}, \qquad k = 0, 1, \ldots, N-1, \tag{1.17}$$

within ϵ; i.e., for all x_k and k, we have $\mu_k^\epsilon(x_k) \in U_k(x_k)$ and

$$\mathop{E}_{w_k}\big\{g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}\big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\big)\big\} \le J_k(x_k) + \epsilon. \tag{1.18}$$

Let $J_k^\epsilon(x_k)$ be the expected cost starting at state x_k at time k, and using the policy $\{\mu_k^\epsilon, \mu_{k+1}^\epsilon, \ldots, \mu_{N-1}^\epsilon\}$. We will show that for all x_k and k, we have

$$J_k(x_k) \le J_k^\epsilon(x_k) \le J_k(x_k) + (N-k)\epsilon, \tag{1.19}$$

$$J_k^*(x_k) \le J_k^\epsilon(x_k) \le J_k^*(x_k) + (N-k)\epsilon, \tag{1.20}$$

$$J_k(x_k) = J_k^*(x_k). \tag{1.21}$$

It is seen using Eq. (1.18) that the inequality (1.19) holds for $k = N-1$. Also since $G_N = J_N$, we have $J_{N-1} = J_{N-1}^*$, which together with Eq. (1.19), implies Eq. (1.20) for $k = N-1$. Thus Eqs. (1.19)-(1.21) hold for index $k = N-1$. Assume that Eqs. (1.19)-(1.21) hold for index $k+1$. We will show that they also hold for index k.

Indeed, we have

$$\begin{aligned}
J_k^\epsilon(x_k) &= \mathop{E}_{w_k}\Big\{g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}^\epsilon\Big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\Big)\Big\} \\
&\le \mathop{E}_{w_k}\Big\{g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}\Big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\Big)\Big\} + (N-k-1)\epsilon \\
&\le J_k(x_k) + \epsilon + (N-k-1)\epsilon \\
&= J_k(x_k) + (N-k)\epsilon,
\end{aligned}$$

where the first equation holds by the definition of J_k^ϵ, the first inequality holds by the induction hypothesis, and the second inequality holds by Eq. (1.18). We also have

$$\begin{aligned}
J_k^\epsilon(x_k) &= \mathop{E}_{w_k}\Big\{g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}^\epsilon\Big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\Big)\Big\} \\
&\ge \mathop{E}_{w_k}\Big\{g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}\Big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\Big)\Big\} \\
&\ge \min_{u_k \in U(x_k)} \mathop{E}_{w_k}\Big\{g_k\big(x_k, u_k, w_k\big) + J_{k+1}\Big(f_k\big(x_k, u_k, w_k\big)\Big)\Big\} \\
&= J_k(x_k),
\end{aligned}$$

where the first inequality holds by the induction hypothesis. Combining the preceding two relations, we see that Eq. (1.19) holds for index k.

For every policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$, we have

$$\begin{aligned}
J_k^\epsilon(x_k) &= \mathop{E}_{w_k}\Big\{g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}^\epsilon\Big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\Big)\Big\} \\
&\le \mathop{E}_{w_k}\Big\{g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}\Big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\Big)\Big\} + (N-k-1)\epsilon \\
&\le J_k(x_k) + \epsilon + (N-k-1)\epsilon \\
&= \min_{u_k \in U(x_k)} \mathop{E}_{w_k}\Big\{g_k\big(x_k, u_k, w_k\big) + J_{k+1}\Big(f_k\big(x_k, u_k, w_k\big)\Big)\Big\} + (N-k)\epsilon \\
&\le \mathop{E}_{w_k}\Big\{g_k\big(x_k, \mu_k(x_k), w_k\big) + J_{\pi^{k+1}}\Big(f_k\big(x_k, \mu_k(x_k), w_k\big)\Big)\Big\} + (N-k)\epsilon \\
&= J_{\pi^k}(x_k) + (N-k)\epsilon,
\end{aligned}$$

where the first inequality holds by the induction hypothesis, and the second inequality holds by Eq. (1.18). Taking the minimum over π^k in the preceding relation, we obtain for all x_k

$$J_k^\epsilon(x_k) \le J_k^*(x_k) + (N-k)\epsilon.$$

We also have by the definition of J_k^*, for all x_k,

$$J_k^*(x_k) \le J_k^\epsilon(x_k).$$

Combining the preceding two relations, we see that Eq. (1.20) holds for index k. Finally, Eq. (1.21) follows from Eqs. (1.19) and (1.20), by taking $\epsilon \to 0$, and the induction is complete.

Note that by using $\epsilon = 0$ in the relation

$$J_0^\epsilon(x_k) \le J_0^*(x_k) + N\epsilon,$$

[cf. Eq. (1.20)], we see that a policy that attains the minimum for all x_k and k in Eq. (1.17) is optimal.

In conclusion, the basic problem has been formulated rigorously, and the DP algorithm has been proved rigorously only when the disturbance spaces $D_0, \ldots, D_{N-1}$ are countable sets, and the expected values of all the cost expressions associated with the problem and the DP algorithm are finite. In the absence of these assumptions, the reader should interpret subsequent results and conclusions as essentially correct but mathematically imprecise statements. In fact, when discussing infinite horizon problems (where the need for precision is greater), we will make the countability assumption explicit.

We note, however, that the advanced reader will have little difficulty in establishing most of our subsequent results concerning specific finite horizon applications, even if the countability assumption is not satisfied. This can be done by using the DP algorithm as a verification theorem. In particular, if one can find within a subset of policies $\tilde{\Pi}$ (such as those satisfying certain measurability restrictions) a policy that attains the minimum in the DP algorithm, then this policy can be readily shown to be optimal within $\tilde{\Pi}$. This result is developed in Exercise 1.29, and can be used by the mathematically oriented reader to establish rigorously many of our subsequent results concerning specific applications. For example, in linear-quadratic problems (Section 3.1) one determines from the DP algorithm a policy in closed form, which is linear in the current state. When w_k can take uncountably many values, it is necessary that admissible policies consist of Borel measurable functions μ_k. Since the linear policy obtained from the DP algorithm belongs to this class, the result of Exercise 1.29 guarantees that this policy is optimal.

For a rigorous mathematical treatment of DP that resolves the associated measurability issues and supplements the present text, we refer to the book [BeS78]. Appendix A of Vol. II provides a more accessible survey. The paper [YuB15] describes some recent related developments relating to the policy iteration method (cf. Section 5.3.2).

1.6 DYNAMIC PROGRAMMING AND MINIMAX CONTROL

The problem of optimal control of uncertain systems has traditionally been treated in a stochastic framework, whereby all uncertain quantities are described by probability distributions, and the expected value of the cost is minimized. However, in many practical situations a stochastic description of the uncertainty may not be available, and one may have information with less detailed structure, such as bounds on the magnitude of the uncertain quantities. In other words, one may know a set within which the uncertain quantities are known to lie, but may not know the corresponding probability distribution. Under these circumstances one may use a minimax approach, whereby the worst possible values of the uncertain quantities within the given set are assumed to occur.

The minimax approach for decision making under uncertainty is described in Appendix F and is contrasted with the expected cost approach, which we have been following so far. In its simplest form, the corresponding decision problem is described by a triplet (Π, W, J), where Π is the set of policies under consideration, W is the set in which the uncertain quantities are known to belong, and $J : \Pi \times W \mapsto [-\infty, +\infty]$ is a given cost function. The objective is to

$$\text{minimize} \quad \max_{w \in W} J(\pi, w)$$

over all $\pi \in \Pi$.

It is possible to formulate a minimax counterpart to the basic problem with perfect state information. This problem is a special case of the abstract minimax problem above, as discussed more fully in Appendix F. Generally, it is unusual for even the simplest special cases of this problem to admit a closed-form solution. However, a computational solution using DP is possible, and our purpose in this section is to describe the corresponding algorithm.

In the framework of the basic problem, consider the case where the disturbances $w_0, w_1, \ldots, w_{N-1}$ do not have a probabilistic description but rather are known to belong to corresponding given sets $W_k(x_k, u_k) \subset D_k$, $k = 0, 1, \ldots, N-1$, which may depend on the current state x_k and control u_k. Consider the problem of finding a policy $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$ with $\mu_k(x_k) \in U_k(x_k)$ for all x_k and k, which minimizes the cost function

$$J_\pi(x_0) = \max_{\substack{w_k \in W_k(x_k, \mu_k(x_k)) \\ k=0,1,\ldots,N-1}} \left[g_N(x_N) + \sum_{k=0}^{N-1} g_k\big(x_k, \mu_k(x_k), w_k\big) \right].$$

The DP algorithm for this problem takes the following form, which resembles the one corresponding to the stochastic basic problem (maximization is used in place of expectation):

$$J_N(x_N) = g_N(x_N), \tag{1.22}$$

$$J_k(x_k) = \min_{u_k \in U(x_k)} \max_{w_k \in W_k(x_k,u_k)} \Big[g_k(x_k,u_k,w_k) + J_{k+1}\big(f_k(x_k,u_k,w_k)\big) \Big]. \tag{1.23}$$

This algorithm can be explained by using a principle of optimality type of argument. In particular, we consider the tail subproblem whereby we are at state x_k at time k, and we wish to minimize the "cost-to-go"

$$\max_{\substack{w_i \in W_i(x_i,\mu_i(x_i)) \\ i=k,k+1,\ldots,N-1}} \left[g_N(x_N) + \sum_{i=k}^{N-1} g_i\big(x_i,\mu_i(x_i),w_i\big) \right],$$

and we argue that if $\pi^* = \{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$ is an optimal policy for the minimax problem, then the truncated policy $\{\mu_k^*, \mu_{k+1}^*, \ldots, \mu_{N-1}^*\}$ is optimal for the tail subproblem. The optimal cost of this subproblem is $J_k(x_k)$, as given by the DP algorithm (1.22)-(1.23). The algorithm expresses the intuitively clear fact that when at state x_k at time k, then regardless of what happened in the past, we should choose u_k that minimizes the worst/maximum value over w_k of the sum of the current stage cost plus the optimal cost of the tail subproblem that starts from the next state.

We will now give a mathematical proof that the DP algorithm (1.22)-(1.23) is valid, and that the optimal cost is equal to $J_0(x_0)$. For this it is necessary to assume that $J_k(x_k) > -\infty$ for all x_k and k. This is analogous to the assumption we made in the preceding section for the validity of the DP algorithm under stochastic disturbances, i.e., that the values $J_k(x_k)$ generated by the DP algorithm are finite for all states x_k and stages k. In the stochastic case the key step of the proof was to bring the minimization over the controls of future stages inside the expectation over the disturbance of the current stage. Similarly, in the minimax case the key step is to bring the minimization over the controls of future stages inside the maximization over the disturbance of the current stage. The following lemma provides the key argument for doing so.

Lemma 1.6.1: Let $f : W \to X$ be a function, and M be the set of all functions $\mu : X \to U$, where W, X, and U are some sets. Then for any functions $G_0 : W \to (-\infty, \infty]$ and $G_1 : X \times U \to (-\infty, \infty]$ such that

$$\min_{u \in U} G_1\big(f(w), u\big) > -\infty, \qquad \text{for all } w \in W,$$

we have

$$\min_{\mu \in M} \max_{w \in W} \Big[G_0(w) + G_1\Big(f(w), \mu\big(f(w)\big)\Big) \Big] = \max_{w \in W} \Big[G_0(w) + \min_{u \in U} G_1\big(f(w), u\big) \Big].$$

Proof: We have for all $\mu \in M$

$$\max_{w \in W} \Big[G_0(w) + G_1\big(f(w), \mu(f(w))\big) \Big] \geq \max_{w \in W} \Big[G_0(w) + \min_{u \in U} G_1\big(f(w), u\big) \Big]$$

and by taking the minimum over $\mu \in M$, we obtain

$$\min_{\mu \in M} \max_{w \in W} \Big[G_0(w) + G_1\big(f(w), \mu(f(w))\big) \Big] \geq \max_{w \in W} \Big[G_0(w) + \min_{u \in U} G_1\big(f(w), u\big) \Big]. \tag{1.24}$$

To show the reverse inequality, for any $\epsilon > 0$, let $\mu_\epsilon \in M$ be such that

$$G_1\big(f(w), \mu_\epsilon(f(w))\big) \leq \min_{u \in U} G_1\big(f(w), u\big) + \epsilon, \qquad \text{for all } w \in W.$$

[Such a μ_ϵ exists because of the assumption $\min_{u \in U} G_1\big(f(w), u\big) > -\infty$.] Then

$$\begin{aligned} \min_{\mu \in M} \max_{w \in W} & \Big[G_0(w) + G_1\big(f(w), \mu(f(w))\big) \Big] \\ & \leq \max_{w \in W} \Big[G_0(w) + G_1\big(f(w), \mu_\epsilon(f(w))\big) \Big] \\ & \leq \max_{w \in W} \Big[G_0(w) + \min_{u \in U} G_1\big(f(w), u\big) \Big] + \epsilon. \end{aligned}$$

Since $\epsilon > 0$ can be taken arbitrarily small, we obtain the reverse to Eq. (1.24), and the desired result follows. **Q.E.D.**

To see how the conclusion of the lemma can fail without the condition

$$\min_{u \in U} G_1\big(f(w), u\big) > -\infty$$

for all w, let u be a scalar, let $w = (w_1, w_2)$ be a two-dimensional vector, and let there be no constraints on u and w ($U = \Re$, $W = \Re \times \Re$, where $\Re$ is the real line). Let also

$$G_0(w) = w_1, \qquad f(w) = w_2, \qquad G_1\big(f(w), u\big) = f(w) + u.$$

Then, for all $\mu \in M$ we have,

$$\max_{w \in W} \Big[G_0(w) + G_1\big(f(w), \mu(f(w))\big) \Big] = \max_{w_1 \in \Re,\ w_2 \in \Re} \big[w_1 + w_2 + \mu(w_2) \big] = \infty,$$

so that

$$\min_{\mu \in M} \max_{w \in W} \Big[G_0(w) + G_1\big(f(w), \mu(f(w))\big) \Big] = \infty.$$

On the other hand,

$$\max_{w \in W} \Big[G_0(w) + \min_{u \in U} G_1\big(f(w), u\big) \Big] = \max_{w_1 \in \Re,\ w_2 \in \Re} \Big[w_1 + \min_{u \in \Re} [w_2 + u] \Big] = -\infty,$$

since $\min_{u\in\Re}[w_2+u]=-\infty$ for all w_2.

We now turn to proving the DP algorithm (1.22)-(1.23). The proof is similar to the one for the DP algorithm for stochastic problems. The optimal cost $J^*(x_0)$ of the problem is given by

$$\begin{aligned} J^*(x_0) = {} & \min_{\mu_0}\cdots\min_{\mu_{N-1}} \max_{w_0\in W[x_0,\mu_0(x_0)]}\cdots\max_{w_{N-1}\in W[x_{N-1},\mu_{N-1}(x_{N-1})]} \\ & \left[\sum_{k=0}^{N-1} g_k\big(x_k,\mu_k(x_k),w_k\big)+g_N(x_N)\right] \\ = {} & \min_{\mu_0}\cdots\min_{\mu_{N-2}}\Bigg[\min_{\mu_{N-1}}\max_{w_0\in W[x_0,\mu_0(x_0)]}\cdots\max_{w_{N-2}\in W[x_{N-2},\mu_{N-2}(x_{N-2})]} \\ & \Bigg[\sum_{k=0}^{N-2} g_k\big(x_k,\mu_k(x_k),w_k\big)+\max_{w_{N-1}\in W[x_{N-1},\mu_{N-1}(x_{N-1})]} \\ & \Big[g_{N-1}\big(x_{N-1},\mu_{N-1}(x_{N-1}),w_{N-1}\big)+J_N(x_N)\Big]\Bigg]\Bigg]. \end{aligned}$$

We can interchange the minimum over μ_{N-1} and the maximum over $w_0,\ldots,w_{N-2}$ by applying Lemma 1.6.1 with the identifications

$$w=(w_0,w_1,\ldots,w_{N-2}),\qquad u=u_{N-1},\qquad f(w)=x_{N-1},$$

$$G_0(w)=\begin{cases}\sum_{k=0}^{N-2} g_k\big(x_k,\mu_k(x_k),w_k\big) & \text{if } w_k\in W_k\big(x_k,\mu_k(x_k)\big) \text{ for all } k,\\ \infty & \text{otherwise,}\end{cases}$$

$$G_1\big(f(w),u\big)=\begin{cases}\hat G_1\big(f(w),u\big) & \text{if } u\in U_{N-1}\big(f(w)\big),\\ \infty & \text{otherwise,}\end{cases}$$

where

$$\begin{aligned}\hat G_1\big(f(w),u\big) = \max_{w_{N-1}\in W_{N-1}\left(f(w),u\right)} & \Big[g_{N-1}\big(f(w),u,w_{N-1}\big) \\ & + J_N\big(f_{N-1}(f(w),u,w_{N-1})\big)\Big],\end{aligned}$$

to obtain

$$\begin{aligned} J^*(x_0) = {} & \min_{\mu_0}\cdots\min_{\mu_{N-2}} \\ & \max_{w_0\in W[x_0,\mu_0(x_0)]}\cdots\max_{w_{N-2}\in W[x_{N-2},\mu_{N-2}(x_{N-2})]} \\ & \left[\sum_{k=0}^{N-2} g_k\big(x_k,\mu_k(x_k),w_k\big)+J_{N-1}(x_{N-1})\right]. \end{aligned}\tag{1.25}$$

The required condition $\min_{u\in U} G_1\big(f(w),u\big)>-\infty$ for all w (required for application of Lemma 1.6.1) is implied by the assumption $J_{N-1}(x_{N-1})>$

$-\infty$ for all x_{N-1}. Now, by working with the expression for $J^*(x_0)$ in Eq. (1.25), and by similarly continuing backwards, with $N-1$ in place of N, etc., after N steps we obtain

$$J^*(x_0) = J_0(x_0),$$

which is the desired relation. The line of argument just given also shows that an optimal policy for the minimax problem can be constructed by minimizing in the right-hand side of the DP Eq. (1.23), similar to the case of the DP algorithm for the stochastic basic problem.

Unfortunately, as mentioned earlier, there are hardly any interesting examples of an analytical, closed-form solution of the DP algorithm (1.22)-(1.23). A computational solution, requires qualitatively comparable effort to the one of the stochastic DP algorithm. Instead of the expectation operation, one must carry out a maximization operation for each x_k and k.

Minimax control problems will be revisited in Chapter 3 in the context of reachability of target sets and target tubes (Section 3.6.2), and in Chapter 6 in the context of model predictive control (Section 6.5.3).

1.7 NOTES, SOURCES, AND EXERCISES

Dynamic programming is a simple mathematical technique that has been used for many years by engineers, mathematicians, and social scientists in a variety of contexts. It was Bellman, however, who realized in the early fifties that DP could be developed (in conjunction with the then appearing digital computer) into a systematic tool for optimization. In his influential books [Bel57], [BeD62], Bellman demonstrated the broad scope of DP and helped streamline its theory.

Following Bellman's works, the mathematical and algorithmic aspects of infinite horizon problems were extensively investigated, extensions to continuous-time problems were formulated and analyzed, and the mathematical issues discussed in Section 1.5 were addressed. In addition, DP was used in a broad variety of applications, ranging from many branches of engineering to statistics, economics, finance, and some of the social sciences. Samples of these applications will be given in subsequent chapters.

A major methodological advance has been the use of various types of approximations in DP methods for large-scale applications, starting in the late 80s. Considerable success has been obtained in a variety of fields, including prominent achievements with programs that have learned how to play games, such as backgammon, Go, chess, and others, at impressive and sometimes above human level. We collectively refer to these methods as "approximate DP"; the name "reinforcement learning" is also often used in artificial intelligence, and the names "neuro-dynamic programming" and "adaptive dynamic programming" are often used in automatic control, with essentially the same meaning. We discuss these methods in Chapter 6 and also, more extensively, in Vol. II of this work. The author's reinforcement learning books [Ber19a], [Ber20a] are focused on approximate DP, using the basic problem of the present chapter as a starting point.

EXERCISES

1.1

Complete the calculations needed to verify that $J_0(1) = 2.7$ and $J_0(2) = 2.818$ in Example 1.3.2.

1.2

Consider the system

$$x_{k+1} = x_k + u_k + w_k, \qquad k = 0, 1, 2, 3,$$

with initial state $x_0 = 5$, and the cost function

$$\sum_{k=0}^{3} (x_k^2 + u_k^2).$$

Apply the DP algorithm for the following three cases:

(a) The control constraint set $U_k(x_k)$ is $\{u \mid 0 \leq x_k + u \leq 5,\ u : \text{ integer}\}$ for all x_k and k, and the disturbance w_k is equal to zero for all k.

(b) The control constraint and the disturbance w_k are as in part (a), but there is in addition a constraint $x_4 = 5$ on the final state. *Hint*: For this problem you need to define a state space for x_4 that consists of just the value $x_4 = 5$, and also to redefine $U_3(x_3)$. Alternatively, you may use a terminal cost $g_4(x_4)$ equal to a very large number for $x_4 \neq 5$.

(c) The control constraint is as in part (a) and the disturbance w_k takes the values -1 and 1 with equal probability $1/2$ for all x_k and u_k, except if $x_k + u_k$ is equal to 0 or 5, in which case $w_k = 0$ with probability 1.

1.3

Suppose we have a machine that is either running or is broken down. If it runs throughout one week, it makes a gross profit of \$100. If it fails during the week, gross profit is zero. If it is running at the start of the week and we perform preventive maintenance, the probability that it will fail during the week is 0.4. If we do not perform such maintenance, the probability of failure is 0.7. However, maintenance will cost \$20. When the machine is broken down at the start of the week, it may either be repaired at a cost of \$40, in which case it will fail during the week with a probability of 0.4, or it may be replaced at a cost of \$150 by a new machine that is guaranteed to run through its first week of operation. Find the optimal repair, replacement, and maintenance policy that maximizes total profit over four weeks, assuming a new machine at the start of the first week.

1.4

A game of the blackjack variety is played by two players as follows: Both players throw a die. The first player, knowing his opponent's result, may stop or may throw the die again and add the result to the result of his previous throw. He then may stop or throw again and add the result of the new throw to the sum of his previous throws. He may repeat this process as many times as he wishes. If his sum exceeds seven (i.e., he busts), he loses the game. If he stops before exceeding seven, the second player takes over and throws the die successively until the sum of his throws is four or higher. If the sum of the second player is over seven, he loses the game. Otherwise the player with the larger sum wins, and in case of a tie the second player wins. The problem is to determine a stopping strategy for the first player that maximizes his probability of winning for each possible initial throw of the second player. Formulate the problem in terms of DP and find an optimal stopping strategy for the case where the second player's initial throw is three. *Hint*: Let $N = 6$ and consider a state space consisting of the following 15 states:

$$x^1 : \text{busted}$$

$$x^{1+i} : \text{already stopped at sum } i \ (1 \leq i \leq 7),$$

$$x^{8+i} : \text{current sum is } i \text{ but the player has not yet stopped } (1 \leq i \leq 7).$$

The optimal strategy is to throw until the sum is four or higher.

1.5 (Computer Assignment)

In the classical game of blackjack the player draws cards knowing only one card of the dealer. The player loses upon reaching a sum of cards exceeding 21. If the player stops before exceeding 21, the dealer draws cards until reaching 17 or higher. The dealer loses upon reaching a sum exceeding 21 or stopping at a lower sum than the player's. If player and dealer end up with an equal sum no one wins. In all other cases the dealer wins. An ace for the player may be counted as a 1 or an 11 as the player chooses. An ace for the dealer is counted as an 11 if this results in a sum from 17 to 21 and as a 1 otherwise. Jacks, queens, and kings count as 10 for both dealer and player. We assume an infinite card deck so the probability of a particular card showing up is independent of earlier cards.

(a) For every possible initial dealer card, calculate the probability that the dealer will reach a sum of 17, 18, 19, 20, 21, or over 21.

(b) Calculate the optimal choice of the player (draw or stop) for each of the possible combinations of dealer's card and player's sum of 12 to 20. Assume that the player's cards do not include an ace.

(c) Repeat part (b) for the case where the player's cards include an ace.

1.6 (Knapsack Problem)

Assume that we have a vessel whose maximum weight capacity is z and whose cargo is to consist of different quantities of N different items. Let v_i denote the value of the ith type of item, w_i the weight of ith type of item, and x_i the number of items of type i that are loaded in the vessel. The problem is to find the most valuable cargo, i.e., to maximize $\sum_{i=1}^{N} x_i v_i$ subject to the constraints $\sum_{i=1}^{N} x_i w_i \leq z$ and $x_i = 0, 1, 2, \ldots$ Formulate this problem in terms of DP.

1.7 (Traveling Repairman Problem)

A repairman must service n sites, which are located along a line and are sequentially numbered $1, 2, \ldots, n$. The repairman starts at a given site s with $1 < s < n$, and is constrained to service only sites that are adjacent to the ones serviced so far, i.e., if he has already serviced sites $i, i+1, \ldots, j$, then he may service next only site $i-1$ (assuming $1 < i$) or site $j+1$ (assuming $j < n$). There is a waiting cost c_i for each time period that site i has remained unserviced and there is a travel cost t_{ij} for servicing site j immediately after servicing site i. Formulate a DP algorithm for finding a minimum cost service schedule.

1.8 (Ordering Matrix Multiplications) (www)

Given a sequence of matrix multiplications

$$M_1 M_2 \cdots M_k M_{k+1} \cdots M_N,$$

where each M_k is a matrix of dimension $n_k \times n_{k+1}$, the order in which multiplications are carried out can make a difference. For example, if $n_1 = 1$, $n_2 = 10$, $n_3 = 1$, and $n_4 = 10$, the calculation $\big((M_1 M_2) M_3\big)$ requires 20 scalar multiplications, but the calculation $\big(M_1 (M_2 M_3)\big)$ requires 200 scalar multiplications (multiplying an $m \times n$ matrix with an $n \times k$ matrix requires mnk scalar multiplications).

(a) Derive a DP algorithm for finding the optimal multiplication order [any order is allowed, including orders that involve multiple partial products each consisting of two or more adjacent matrices, e.g., $\big((M_1 M_2)(M_3 M_4)\big)$]. Solve the problem for $N = 3$, $n_1 = 2$, $n_2 = 10$, $n_3 = 5$, and $n_4 = 1$.

(b) Derive a DP algorithm for finding the optimal multiplication order within the class of orders where at each step, we maintain only one partial product that consists only of adjacent matrices, e.g., $\Big(\big(M_1 (M_2 M_3)\big) M_4\Big)$.

1.9 (Paragraphing Problem)

The paragraphing problem deals with breaking up a sequence of N words of given lengths into lines of length A. Let $w_1, \ldots, w_N$ be the words and let $L_1, \ldots, L_N$ be their lengths. In a simple version of the problem, words are separated by blanks whose ideal width is b, but blanks can stretch or shrink if necessary, so that a line $w_i, w_{i+1}, \ldots, w_{i+k}$ has length exactly A. The cost associated with the line is $(k+1)|b' - b|$, where $b' = (A - L_i - \cdots - L_{i+k})/(k+1)$ is the actual average width of the blanks, except if we have the last line ($N = i + k$), in which case the cost is zero when $b' \geq b$. Formulate a DP algorithm for finding the minimum cost separation. *Hint*: Consider the subproblems of optimally separating $w_i, \ldots, w_N$ for $i = 1, \ldots, N$.

1.10 (Interval Scheduling)

We have N intervals labeled $1, \ldots, N$. The ith interval has start point y_i, end point z_i, and value v_i. We want to select a subset of these intervals that has maximum total value, and such that no pair overlaps. Formulate a DP algorithm to solve this problem. *Hint*: Suppose the intervals are ordered so that $z_1 \leq \cdots \leq z_N$. Let the number of periods be N and the states be $z_1, \ldots, z_N$. Let also the optimal value at state z_k be the maximal value over nonoverlapping intervals whose start time is greater than z_i.

1.11

Consider a smaller version of a popular puzzle game. Three square tiles numbered 1, 2, and 3 are placed in a 2×2 grid with one space left empty. The two tiles adjacent to the empty space can be moved into that space, thereby creating new configurations. Use a DP argument to answer the question whether it is possible to generate a given configuration starting from any other configuration.

1.12

From a pile of eleven matchsticks, two players take turns removing one or four sticks. The player who removes the last stick wins. Use a DP argument to show that there is a winning strategy for the player who plays first.

1.13 (Counterfeit Coin Problem)

We are given six coins, one of which is counterfeit and is known to have different weight than the rest. Construct a strategy to find the counterfeit coin using a two-pan scale in a minimum average number of tries. *Hint*: There are two initial decisions that make sense: (1) test two of the coins against two others, and (2) test one of the coins against one other.

1.14 (Multiplicative Cost)

In the framework of the basic problem, consider the case where the cost has the multiplicative form

$$\mathop{E}_{\substack{w_k \\ k=0,1,\ldots,N-1}} \big\{ g_N(x_N) \cdot g_{N-1}(x_{N-1}, u_{N-1}, w_{N-1}) \cdots g_0(x_0, u_0, w_0) \big\}.$$

Develop a DP-like algorithm for this problem assuming that $g_k(x_k, u_k, w_k) \geq 0$ for all x_k, u_k, w_k, and k.

1.15

Consider a device consisting of N stages connected in series, where each stage consists of a particular component. The components are subject to failure, and to increase the reliability of the device duplicate components are provided. For $j = 1, 2, \ldots, N$, let $(1 + m_j)$ be the number of components for the jth stage, let $p_j(m_j)$ be the probability of successful operation of the jth stage when $(1 + m_j)$ components are used, and let c_j denote the cost of a single component at the jth stage. Formulate in terms of DP the problem of finding the number of components at each stage that maximize the reliability of the device expressed by

$$p_1(m_1) \cdot p_2(m_2) \cdots p_N(m_N),$$

subject to the cost constraint $\sum_{j=1}^N c_j m_j \leq A$, where $A > 0$ is given.

1.16 (www)

An innkeeper charges a different rate for a room as the day progresses, depending on whether he has many or few vacancies. His objective is to maximize his expected total income during the day. Let x be the number of empty rooms at the start of the day, and let y be the number of customers that will ask for a room in the course of the day. We assume (somewhat unrealistically) that the innkeeper knows y with certainty, and upon arrival of a customer, quotes one of m prices r_i, $i = 1, \ldots, m$, where $0 < r_1 \leq r_2 \leq \cdots \leq r_m$. A quote of a rate r_i is accepted with probability p_i and is rejected with probability $1 - p_i$, in which case the customer departs, never to return during that day.

(a) Formulate this as a problem with y stages and show that the maximal expected income, as a function of x and y, satisfies the recursion

$$J(x, y) = \max_{i=1,\ldots,m} \Big[p_i \big(r_i + J(x-1, y-1) \big) + (1 - p_i) J(x, y-1) \Big],$$

for all $x \geq 1$ and $y \geq 1$, with initial conditions

$$J(x, 0) = J(0, y) = 0, \qquad \text{for all } x \text{ and } y.$$

Assuming that the product $p_i r_i$ is monotonically nondecreasing with i, and that p_i is monotonically nonincreasing with i, show that the innkeeper should always charge the highest rate r_m.

(b) Consider a variant of the problem where each arriving customer, with probability p_i, offers a price r_i for a room, which the innkeeper may accept or reject. In the latter case the customer departs, never to return during that day. Show that an appropriate DP algorithm is

$$J(x,y) = \sum_{i=1}^{m} p_i \max\bigl[r_i + J(x-1, y-1),\ J(x, y-1)\bigr],$$

with initial conditions

$$J(x,0) = J(0,y) = 0, \qquad \text{for all } x \text{ and } y.$$

Show also that for given x and y it is optimal to accept a customer's offer if it is larger than some threshold $\overline{r}(x,y)$. *Hint*: This part is related to DP for uncontrollable state components (cf. Section 1.4).

1.17 (Investing in a Stock) www

An investor observes at the beginning of each period k the price x_k of a stock and decides whether to buy 1 unit, sell 1 unit, or do nothing. There is a transaction cost c for buying or selling. The stock price can take one of n different values $v^1, \ldots, v^n$ and the transition probabilities

$$p_{ij}^k = P\{x_{k+1} = v^j \mid x_k = v^i\}$$

are known. The investor wants to maximize the total worth of his stock at a fixed final period N minus his investment costs from period 0 to period $N-1$ (revenue from a sale is viewed as negative cost). We assume that the function

$$P_k(x) = E\{x_N \mid x_k = x\} - x$$

is monotonically nonincreasing as a function of x; i.e., the expected profit from a purchase is a nonincreasing function of the purchase price.

(a) Assume that the investor starts with N or more units of stock and an unlimited amount of cash, so that a purchase or sale decision is possible at each period regardless of the past decisions and the current price. For every period k, let $\underline{x}_k$ be the largest value of $x \in \{v^1, \ldots, v^n\}$ such that $P_k(x) > c$, and let $\overline{x}_k$ be the smallest value of $x \in \{v^1, \ldots, v^n\}$ such that $P_k(x) < -c$. Show that it is optimal to buy if $x_k \le \underline{x}_k$, sell if $\overline{x}_k \le x_k$, and do nothing otherwise. *Hint*: Formulate the problem as one of maximizing

$$E\left\{\sum_{k=0}^{N-1} \bigl(u_k P_k(x_k) - c\,|u_k|\bigr)\right\},$$

where $u_k \in \{-1, 0, 1\}$.

(b) Formulate an efficient DP algorithm for the case where the investor starts with less than N units of stock and an unlimited amount of cash. Show that it is still optimal to buy if $x_k \leq \underline{x}_k$ and it is still not optimal to sell if $x_k < \overline{x}_k$. Could it be optimal to buy at any prices x_k greater than $\underline{x}_k$?

(c) Consider the situation where the investor initially has N or more units of stock and there is a constraint that for any time k the number of purchase decisions up to k should not exceed the number of sale decisions up to k by more that a given fixed number m (this models approximately the situation where the investor has a limited initial amount of cash). Formulate an efficient DP algorithm for this case. Show that it is still optimal to sell if $\overline{x}_k \leq x_k$ and it is still not optimal to buy if $\underline{x}_k < x_k$.

(d) Consider the situation where there are restrictions on both the initial amount of stock as in part (b), and the number of purchase decisions as in part (c). Derive a DP algorithm for this problem.

(e) How would the analysis of (a)-(d) be affected if cash is invested at a given fixed interest rate?

1.18 (Regular Polygon Theorem) www

According to a famous theorem (attributed to the ancient Greek geometer Zenodorus), of all N-side polygons inscribed in a given circle, those that are regular (all sides are equal) have maximal area.

(a) Prove the theorem by applying DP to a suitable problem involving sequential placement of N points in the circle.

(b) Use DP to solve the problem of placing a given number of points on a subarc of the circle, so as to maximize the area of the polygon whose vertices are these points, the endpoints of the subarc, and the center of the circle.

1.19 (Inscribed Polygon of Maximal Perimeter)

Consider the problem of inscribing an N-side polygon in a given circle, so that the polygon has maximal perimeter.

(a) Formulate the problem as a DP problem involving sequential placement of N points in the circle.

(b) Use DP to show that the optimal polygon is regular (all sides are equal).

1.20 www

A decision maker must continually choose between two activities over a time interval $[0, T]$. Choosing activity i at time t, where $i = 1, 2$, earns reward at a rate $g_i(t)$, and every switch between the two activities costs $c > 0$. Thus, for example, the reward for starting with activity 1, switching to 2 at time t_1, and switching back to 1 at time $t_2 > t_1$ earns total reward

$$\int_0^{t_1} g_1(t)\,dt + \int_{t_1}^{t_2} g_2(t)\,dt + \int_{t_2}^{T} g_1(t)\,dt - 2c.$$

We want to find a set of switching times that maximize the total reward. Assume that the function $g_1(t) - g_2(t)$ changes sign a finite number of times in the interval $[0, T]$. Formulate the problem as a finite horizon problem and write the corresponding DP algorithm. See Shreve [Shr81] for a fuller development of this problem.

1.21

A farmer annually producing x_k units of a certain crop stores $(1 - u_k)x_k$ units of his production, where $0 \leq u_k \leq 1$, and invests the remaining $u_k x_k$ units, thus increasing the next year's production to a level x_{k+1} given by

$$x_{k+1} = x_k + w_k u_k x_k, \qquad k = 0, 1, \ldots, N-1.$$

The scalars w_k are independent random variables with identical probability distributions that do not depend either on x_k or u_k. Furthermore, $E\{w_k\} = \overline{w} > 0$. The problem is to find the optimal investment policy that maximizes the total expected product stored over N years

$$\mathop{E}_{\substack{w_k \\ k=0,1,\ldots,N-1}} \left\{ x_N + \sum_{k=0}^{N-1} (1 - u_k) x_k \right\}.$$

Show the optimality of the following policy that consists of constant functions:

(a) If $\overline{w} > 1$, $\mu_0^*(x_0) = \cdots = \mu_{N-1}^*(x_{N-1}) = 1$.

(b) If $0 < \overline{w} < 1/N$, $\mu_0^*(x_0) = \cdots = \mu_{N-1}^*(x_{N-1}) = 0$.

(c) If $1/N \leq \overline{w} \leq 1$,

$$\mu_0^*(x_0) = \cdots = \mu_{N-\overline{k}-1}(x_{N-\overline{k}-1}) = 1,$$

$$\mu_{N-\overline{k}}^*(x_{N-\overline{k}}) = \cdots = \mu_{N-1}^*(x_{N-1}) = 0,$$

where $\overline{k}$ is such that $1/(\overline{k}+1) < \overline{w} \leq 1/\overline{k}$.

1.22

Let x_k denote the number of educators in a certain country at time k and let y_k denote the number of research scientists at time k. New scientists (potential educators or research scientists) are produced during the kth period by educators at a rate γ_k per educator, while educators and research scientists leave the field due to death, retirement, and transfer at a rate δ_k. The scalars γ_k, $k = 0, 1, \ldots, N-1$, are independent identically distributed random variables taking values within a closed and bounded interval of positive numbers. Similarly δ_k, $k = 0, 1, \ldots, N-1$, are independent identically distributed and take values in an interval $[\delta, \delta']$ with $0 < \delta \leq \delta' < 1$. By means of incentives, a science policy maker can determine the proportion u_k of new scientists produced at time k who become educators. Thus, the number of research scientists and educators evolves according to the equations

$$x_{k+1} = (1 - \delta_k)x_k + u_k \gamma_k x_k,$$

$$y_{k+1} = (1 - \delta_k)y_k + (1 - u_k)\gamma_k x_k.$$

The initial numbers x_0, y_0 are known, and it is required to find a policy

$$\big\{\mu_0^*(x_0, y_0), \ldots, \mu_{N-1}^*(x_{N-1}, y_{N-1})\big\}$$

with

$$0 < \alpha \leq \mu_k^*(x_k, y_k) \leq \beta < 1, \qquad \text{for all } x_k, y_k, \text{and } k,$$

which maximizes $E_{\gamma_k, \delta_k}\{y_N\}$ (i.e., the expected final number of research scientists after N periods). The scalars α and β are given.

(a) Show that the cost-to-go functions $J_k(x_k, y_k)$ are linear; i.e., for some scalars ξ_k, ζ_k,

$$J_k(x_k, y_k) = \xi_k x_k + \zeta_k y_k.$$

(b) Derive an optimal policy $\{\mu_0^*, \ldots, \mu_{N-1}^*\}$ under the assumption

$$E\{\gamma_k\} > E\{\delta_k\}$$

and show that this optimal policy can consist of constant functions.

(c) Assume that the proportion of new scientists who become educators at time k is $u_k + \epsilon_k$ (rather than u_k), where ϵ_k are identically distributed independent random variables that are also independent of γ_k, δ_k and take values in the interval $[-\alpha, 1-\beta]$. Derive the form of the cost-to-go functions and the optimal policy.

1.23 (Discounted Cost per Stage)

In the framework of the basic problem, consider the case where the cost is of the form

$$\mathop{E}_{\substack{w_k \\ k=0,1,\ldots,N-1}} \left\{ \alpha^N g_N(x_N) + \sum_{k=0}^{N-1} \alpha^k g_k(x_k, u_k, w_k) \right\},$$

where α is a discount factor with $0 < \alpha < 1$. Show that an alternate form of the DP algorithm is given by

$$V_N(x_N) = g_N(x_N),$$

$$V_k(x_k) = \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ g_k(x_k, u_k, w_k) + \alpha V_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\}.$$

1.24 (Exponential Cost Function)

In the framework of the basic problem, consider the case where the cost is of the form

$$\mathop{E}_{\substack{w_k\\ k=0,1,\ldots,N-1}} \left\{ \exp\left(g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right) \right\}.$$

(a) Show that the optimal cost and an optimal policy can be obtained from the DP-like algorithm

$$J_N(x_N) = \exp\big(g_N(x_N)\big),$$

$$J_k(x_k) = \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ J_{k+1}\big(f_k(x_k, u_k, w_k)\big) \exp\big(g_k(x_k, u_k, w_k)\big) \Big\}.$$

(b) Define the functions $V_k(x_k) = \ln J_k(x_k)$. Assume also that g_k is a function of x_k and u_k only (and not of w_k). Show that the above algorithm can be rewritten as

$$V_N(x_N) = g_N(x_N),$$

$$V_k(x_k) = \min_{u_k \in U_k(x_k)} \left\{ g_k(x_k, u_k) + \ln \mathop{E}_{w_k} \Big\{ \exp\Big(V_{k+1}\big(f_k(x_k, u_k, w_k)\big)\Big) \Big\} \right\}.$$

Note: The exponential is an example of a *risk-sensitive cost function* that can be used to encode a preference for policies with a small variance of the cost $g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k)$. The associated problems have a lot of interesting properties, which are discussed in several sources, e.g., [DeR79], [Whi90], [FeM94], [JBE94], [BaB95], [Bas00], [Pat01], [Ber16b].

1.25 (Terminating Process)

In the framework of the basic problem, consider the case where the system evolution terminates at time i when a given value $\overline{w}_i$ of the disturbance at time i occurs, or when a termination decision u_i is made by the controller. If termination occurs at time i, the resulting cost is

$$T + \sum_{k=0}^{i} g_k(x_k, u_k, w_k),$$

where T is a termination cost. If the process has not terminated up to the final time N, the resulting cost is $g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k)$. Reformulate the problem into the framework of the basic problem. *Hint*: Augment the state space with a special termination state.

1.26

Alexei plays a game that starts with a deck with b "black" cards and r "red" cards. At each time period he draws a random card and decides between the following two options:

(1) Without looking at the card, "predict" that it is black, in which case he wins the game if the prediction is correct and loses if the prediction is incorrect.

(2) "Discard" the card, after looking at its color, and continue the game with one card less.

If the deck has only black cards he wins the game, while if the deck has only red cards he loses the game. Alexei wants to find a policy that maximizes his probability of a win.

(a) Formulate Alexei's problem into the format of the finite-horizon basic problem with perfect state information. Identify states, controls, and disturbances, and write the DP algorithm.

(b) Use induction to show that the optimal probability of a win starting with b black cards and r red cards is $\dfrac{b}{b+r}$.

(c) Characterize the optimal policies.

(d) Suppose that Alexei is given the additional option to randomize his decision at each time period. In particular, he may choose a probability $p \in [0,1]$, flip a coin that has probability of head equal to p, and decide upon option 1 or 2 above depending on the outcome of the flip. What would then be the optimal policies?

1.27 (Semilinear Systems) (www)

Consider a problem involving the system

$$x_{k+1} = A_k x_k + f_k(u_k) + w_k,$$

where $x_k \in \Re^n$, f_k are given functions, and A_k and w_k are random $n \times n$ matrices and n-vectors, respectively, with given probability distributions that do not depend on x_k, u_k or prior values of A_k and w_k. Assume that the cost function is linear in the states and has the form

$$\mathop{E}_{\substack{A_k, w_k \\ k=0,1,\ldots,N-1}} \left\{ c_N' x_N + \sum_{k=0}^{N-1} \Big(c_k' x_k + g_k\big(\mu_k(x_k)\big) \Big) \right\},$$

where c_k are given vectors and g_k are given functions. Show that if the optimal cost for this problem is finite and the control constraint sets $U_k(x_k)$ are independent of x_k, then the cost-to-go functions of the DP algorithm are affine (linear plus constant). Assuming that there is at least one optimal policy, show that there exists an optimal policy that is open-loop, i.e., $\mu_k^*(x_k) = \text{constant}$ for all $x_k \in \Re^n$.

1.28 (Monotonicity Property of DP) (www)

An evident, yet very important property of the DP algorithm is that if the terminal cost g_N is changed to a uniformly larger cost $\overline{g}_N$ [i.e., $g_N(x_N) \leq \overline{g}_N(x_N)$ for all x_N], then the last stage cost-to-go $J_{N-1}(x_{N-1})$ will be uniformly increased. More generally, given two functions J_{k+1} and $\bar{J}_{k+1}$ with $J_{k+1}(x_{k+1}) \leq \bar{J}_{k+1}(x_{k+1})$ for all x_{k+1}, we have, for all x_k and $u_k \in U_k(x_k)$,

$$\begin{aligned} &\mathop{E}_{w_k}\Big\{g_k(x_k,u_k,w_k) + J_{k+1}\big(f_k(x_k,u_k,w_k)\big)\Big\} \\ &\qquad\qquad \leq \mathop{E}_{w_k}\Big\{g_k(x_k,u_k,w_k) + \bar{J}_{k+1}\big(f_k(x_k,u_k,w_k)\big)\Big\}. \end{aligned}$$

Suppose now that in the basic problem the system and cost are time invariant; i.e., $S_k \equiv S$, $C_k \equiv C$, $D_k \equiv D$, $f_k \equiv f$, $U_k \equiv U$, $P_k \equiv P$, and $g_k \equiv g$ for some S, C, D, f, U, P, and g. Use induction to show that if in the DP algorithm we have $J_{N-1}(x) \leq J_N(x)$ for all $x \in S$, then

$$J_k(x) \leq J_{k+1}(x), \qquad \text{for all } x \in S \text{ and } k.$$

Similarly, if we have $J_{N-1}(x) \geq J_N(x)$ for all $x \in S$, then

$$J_k(x) \geq J_{k+1}(x), \qquad \text{for all } x \in S \text{ and } k.$$

1.29 (DP Algorithm for Minimization over a Subset of Policies)

This exercise is primarily of theoretical interest (see the discussion at the end of Section 1.5), but also relates to situations where we can guess that an optimal policy may be found within a special class of policies. Consider a variation of the basic problem whereby we want to find

$$\min_{\pi \in \tilde{\Pi}} J_\pi(x_0),$$

where $\tilde{\Pi}$ is some given *subset* of the set of sequences $\{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$ of functions $\mu_k : S_k \to C_k$ with $\mu_k(x_k) \in U_k(x_k)$ for all $x_k \in S_k$. Assume that for every $\pi = \{\mu_0, \ldots, \mu_{N-1}\} \in \tilde{\Pi}$, the sequence of cost-to-go functions $\tilde{J}_{\pi,k}$, $k = 0, \ldots, N$, generated by

$$\tilde{J}_{\pi,N}(x_N) = g_N(x_N),$$

$$\tilde{J}_{\pi,k}(x_k) = \mathop{E}_{w_k}\Big\{g_k\big(x_k,\mu_k(x_k),w_k\big) + \tilde{J}_{\pi,k+1}\big(f_k(x_k,\mu_k(x_k),w_k)\big)\Big\},$$

is well-defined in the sense that the functions $\tilde{J}_{\pi,k}$ are real-valued, and that the expected value in the preceding equation is well-defined and finite. Suppose also that

$$\tilde{\pi} = \{\tilde{\mu}_0, \tilde{\mu}_1, \ldots, \tilde{\mu}_{N-1}\}$$

is a policy that belongs to $\tilde{\Pi}$ and attains the minimum in the DP algorithm, in the sense that for all x_k and $k = 0, \ldots, N-1$, we have

$$\begin{aligned} &\mathop{E}_{w_k} \Big\{ g_k\big(x_k, \tilde{\mu}_k(x_k), w_k\big) + J_{\tilde{\pi},k+1}\big(f_k(x_k, \tilde{\mu}_k(x_k), w_k)\big)\Big\} \\ &\qquad = \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ g_k(x_k, u_k, w_k) + J_{\tilde{\pi},k+1}\big(f_k(x_k, u_k, w_k)\big)\Big\}. \end{aligned}$$

Show that $\tilde{\pi}$ is optimal within $\tilde{\Pi}$ in the sense that $J_{\tilde{\pi},0}(x_0) \le J_{\pi,0}(x_0)$ for all $\pi \in \tilde{\Pi}$ and states x_0. *Hint*: Use backwards induction to show that $J_{\tilde{\pi},k}(x_k) \le J_{\pi,k}(x_k)$ for all $\pi \in \tilde{\Pi}$, k, and states x_k.

1.30 (Post-Decision States)

Consider the basic problem and assume that the system equation has a special structure whereby from state x_k after applying u_k we move to an intermediate "post-decision state" $y_k = p_k(x_k, u_k)$ at cost $g_k(x_k, u_k)$. Then from y_k we move at no cost to the new state x_{k+1} according to

$$x_{k+1} = h_k(y_k, w_k),$$

where the distribution of the disturbance w_k depends only on y_k, and not on prior disturbances, states, and controls. The purpose of this exercise is to show that it is possible to exploit the structure of the problem to execute the DP algorithm more efficiently. Denote by $J_k(x_k)$ the optimal cost-to-go starting at time k from state x_k, and by $V_k(y_k)$ the optimal cost-to-go starting at time k from post-decision state y_k.

(a) Show that a DP algorithm that generates only J_k is given by

$$J_k(x_k) = \min_{u_k \in U_k(x_k)} \Big[g(x_k, u_k) + E_{w_k}\big\{ J_{k+1}\big(h_k(p_k(x_k, u_k), w_k)\big)\big\}\Big].$$

(b) Show that a DP algorithm that generates both J_k and V_k is given by

$$J_k(x_k) = \min_{u_k \in U_k(x_k)} \bigg[g(x_k, u_k) + V_k\big(p_k(x_k, u_k)\big)\bigg],$$

$$V_k(y_k) = E_{w_k}\bigg\{ J_{k+1}\big(h_k(y_k, w_k), w_k\big)\bigg\}.$$

(c) Show that a DP algorithm that generates only V_k for all k is given by

$$\begin{aligned} V_k(y_k) = E_{w_k}\bigg\{ &\min_{u_{k+1} \in U_{k+1}(h_k(y_k, w_k))} \big[g_{k+1}(h_k(y_k, w_k), u_{k+1}) \\ &\qquad + V_{k+1}(p_{k+1}(h_k(y_k, w_k), u_{k+1}))\big]\bigg\}. \end{aligned}$$

References

[ABC65] Atkinson, R. C., Bower, G. H., and Crothers, E. J., 1965. An Introduction to Mathematical Learning Theory, Wiley, N. Y.

[ABF93] Arapostathis, A., Borkar, V., Fernandez-Gaucherand, E., Ghosh, M., and Marcus, S., 1993. "Discrete-Time Controlled Markov Processes with Average Cost Criterion: A Survey," SIAM J. on Control and Optimization, Vol. 31, pp. 282-344.

[ABG49] Arrow, K. J., Blackwell, D., and Girshick, M. A., 1949. "Bayes and Minimax Solutions of Sequential Design Problems," Econometrica, Vol. 17, pp. 213-244.

[AGK77] Athans, M., Ku, R., and Gershwin, S. B., 1977. "The Uncertainty Threshold Principle," IEEE Trans. on Automatic Control, Vol. AC-22, pp. 491-495.

[AHM51] Arrow, K. J., Harris, T., and Marschack, J., 1951. "Optimal Inventory Policy," Econometrica, Vol. 19, pp. 250-272.

[AKS58] Arrow, K. J., Karlin, S., and Scarf, H., 1958. Studies in the Mathematical Theory of Inventory and Production, Stanford Univ. Press, Stanford, CA.

[Abr90] Abramson, B., 1990. "Expected-Outcome: A General Model of Static Evaluation," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, pp. 182-193.

[AdG86] Adams, M., and Guillemin, V., 1986. Measure Theory and Probability, Wadsworth and Brooks, Monterey, CA.

[AnH14] Antunes, D., and Heemels, W.P.M.H., 2014. "Rollout Event-Triggered Control: Beyond Periodic Control Performance," IEEE Transactions on Automatic Control, Vol. 59, pp. 3296-3311.

[AnM79] Anderson, B. D. O., and Moore, J. B., 1979. Optimal Filtering, Prentice-Hall, Englewood Cliffs, N. J.

[AoL69] Aoki, M., and Li, M. T., 1969. "Optimal Discrete-Time Control Systems with Cost for Observation," IEEE Trans. Automatic Control, Vol. AC-14, pp. 165-175.

[AsW94] Aström, K. J., and Wittenmark, B., 1994. Adaptive Control, 2nd Edition, Prentice-Hall, Englewood Cliffs, N. J.

[Ash70] Ash, R. B., 1970. Basic Probability Theory, Wiley, N. Y.

[Ash72] Ash, R. B., 1972. Real Analysis and Probability, Academic Press, N. Y.

[Ast83] Aström, K. J., 1983. "Theory and Applications of Adaptive Control – A Survey," Automatica, Vol. 19, pp. 471-486.

[AtF66] Athans, M., and Falb, P., 1966. Optimal Control, McGraw-Hill, N. Y.

[BBC11] Bertsimas, D., Brown, D. B., and Caramanis, C., 2011. “Theory and Applications of Robust Optimization,” SIAM Review, Vol. 53, pp. 464-501.

[BBD10] Busoniu, L., Babuska, R., De Schutter, B., and Ernst, D., 2010. Reinforcement Learning and Dynamic Programming Using Function Approximators, CRC Press, N. Y.

[BBG13] Bertazzi, L., Bosco, A., Guerriero, F., and Lagana, D., 2013. “A Stochastic Inventory Routing Problem with Stock-Out,” Transportation Research, Part C, Vol. 27, pp. 89-107.

[BBM17] Borelli, F., Bemporad, A., and Morari, M., 2017. Predictive Control for Linear and Hybrid Systems, Cambridge Univ. Press, Cambridge, UK.

[BCN18] Bottou, L., Curtis, F. E., and Nocedal, J., 2018. “Optimization Methods for Large-Scale Machine Learning,” SIAM Review, Vol. 60, pp. 223-311.

[BGM95] Bertsekas, D. P., Guerriero, F., and Musmanno, R., 1995. “Parallel Shortest Path Methods for Globally Optimal Trajectories,” High Performance Computing: Technology, Methods, and Applications, (J. Dongarra et al., Eds.), Elsevier.

[BGM96] Bertsekas, D. P., Guerriero, F., and Musmanno, R., 1996. “Parallel Label Correcting Methods for Shortest Paths,” J. Optimization Theory Appl., Vol. 88, 1996, pp. 297-320.

[BKB20] Bhattacharya, S., Kailas, S., Badyal, S., Gil, S., and Bertsekas, D. P., 2020. “Multiagent Rollout and Policy Iteration for POMDP with Application to Multi-Robot Repair Problems,” Working paper, Arizona State University, and Harvard University.

[BKM05] de Boer, P. T., Kroese, D. P., Mannor, S., and Rubinstein, R. Y. 2005. “A Tutorial on the Cross-Entropy Method,” Annals of Operations Research, Vol. 134, pp. 19-67.

[BMO14] Boyd, S., Mueller, M. T., O’Donoghue, B., and Wang, Y., 2014. “Performance Bounds and Suboptimal Policies for Multi-Period Investment,” Foundations and Trends in Optimization, Vol. 1, pp. 1-72.

[BMS99] Boltyanski, V., Martini, H., and Soltan, V., 1999. Geometric Methods and Optimization Problems, Kluwer, Boston.

[BNO03] Bertsekas, D. P., Nedić, A., and Ozdaglar, A. E., 2003. Convex Analysis and Optimization, Athena Scientific, Belmont, MA.

[BTW97] Bertsekas, D. P., Tsitsiklis, J. N., and Wu, C., 1997. “Rollout Algorithms for Combinatorial Optimization,” Heuristics, Vol. 3, pp. 245-262.

[BYB94] Bradtke, S. J., Ydstie, B. E., and Barto, A. G., 1994. “Adaptive Linear Quadratic Control Using Policy Iteration,” Proc. IEEE American Control Conference, Vol. 3, pp. 3475-3479.

[BaB95] Basar, T., and Bernhard, P., 1995. H_∞ Optimal Control and Related Minimax Design Problems: A Dynamic Game Approach, Birkhuser, Boston, MA.

[Bai93] Baird, L. C., 1993. “Advantage Updating,” Report WL-TR-93-1146, Wright Patterson AFB, OH.

[Bai94] Baird, L. C., 1994. “Reinforcement Learning in Continuous Time: Advantage Updating,” International Conf. on Neural Networks, Orlando, Fla.

[Bar81] Bar-Shalom, Y., 1981. “Stochastic Dynamic Programming: Caution and Probing,” IEEE Trans. on Automatic Control, Vol. AC-26, pp. 1184-1195.

[Bas91] Basar, T., 1991. “Optimum Performance Levels for Minimax Filters, Predictors, and Smoothers,” Systems and Control Letters, Vol. 16, pp. 309-317.

[Bas00] Basar, T., 2000. “Risk-Averse Designs: From Exponential Cost to Stochastic Games,” In T. E. Djaferis and I. C. Schick, (Eds.), System Theory: Modeling, Analysis and Control, Kluwer, Boston, pp. 131-144.

[BeC99] Bertsekas, D. P., and Castanon, D. A., 1999. “Rollout Algorithms for Stochastic Scheduling Problems,” Heuristics, Vol. 5, pp. 89-108.

[BeC04] Bertsekas, D. P., and Castanon, D. A., 2004. Unpublished Collaboration.

[BeC08] Besse, C., and Chaib-draa, B., 2008. “Parallel Rollout for Online Solution of DEC-POMDPs,” Proc. of 21st International FLAIRS Conference, pp. 619-624.

[BeD62] Bellman, R., and Dreyfus, S., 1962. Applied Dynamic Programming, Princeton Univ. Press, Princeton, N. J.

[BeG92] Bertsekas, D. P., and Gallager, R. G., 1992. Data Networks, 2nd Edition, Prentice-Hall, Englewood Cliffs, N. J.

[BeL14] Beyme, S., and Leung, C., 2014. “Rollout Algorithm for Target Search in a Wireless Sensor Network,” 80th Vehicular Technology Conference (VTC2014), IEEE, pp. 1-5.

[BeI96] Bertsekas, D. P., and Ioffe, S., 1996. “Temporal Differences-Based Policy Iteration and Applications in Neuro-Dynamic Programming,” Lab. for Info. and Decision Systems Report LIDS-P-2349, Massachusetts Institute of Technology.

[BeN01] Ben-Tal, A., and Nemirovski, A., 2001. Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications, SIAM, Phila., PA

[BeP03] Bertsimas, D., and Popescu, I., 2003. “Revenue Management in a Dynamic Network Environment,” Transportation Science, Vol. 37, pp. 257-277.

[BeR71a] Bertsekas, D. P., and Rhodes, I. B., 1971. “On the Minimax Reachability of Target Sets and Target Tubes,” Automatica, Vol. 7, pp. 233-247.

[BeR71b] Bertsekas, D. P., and Rhodes, I. B., 1971. “Recursive State Estimation for a Set-Membership Description of the Uncertainty,” IEEE Trans. Automatic Control, Vol. AC-16, pp. 117-128.

[BeR73] Bertsekas, D. P., and Rhodes, I. B., 1973. “Sufficiently Informative Functions and the Minimax Feedback Control of Uncertain Dynamic Systems,” IEEE Trans. Automatic Control, Vol. AC-18, pp. 117-124.

[BeS78] Bertsekas, D. P., and Shreve, S. E., 1978. Stochastic Optimal Control: The Discrete Time Case, Academic Press, N. Y.; republished by Athena Scientific, Belmont, MA, 1996 (can be downloaded from the author’s website).

[BeS18] Bertazzi, L., and Secomandi, N., 2018. “Faster Rollout Search for the Vehicle Routing Problem with Stochastic Demands and Restocking,” European J. of Operational Research, Vol. 270, pp.487-497.

[BeT89] Bertsekas, D. P., and Tsitsiklis, J. N., 1989. Parallel and Distributed Computation: Numerical Methods, Prentice-Hall, Englewood Cliffs, N. J.; republished by Athena Scientific, Belmont, MA, 1997.

[BeT91] Bertsekas, D. P., and Tsitsiklis, J. N., 1991. “An Analysis of Stochastic Shortest Path Problems,” Math. Operations Res., Vol. 16, pp. 580-595.

[BeT96] Bertsekas, D. P., and Tsitsiklis, J. N., 1996. Neuro-Dynamic Programming, Athena Scientific, Belmont, MA.

[BeT97] Bertsimas, D., and Tsitsiklis, J. N., 1997. Introduction to Linear Optimization, Athena Scientific, Belmont, MA.

[BeT00] Bertsekas, D. P., and Tsitsiklis, J. N., 2000. "Gradient Convergence of Gradient Methods with Errors," SIAM J. on Optimization, Vol. 36, pp. 627-642.

[BeT08] Bertsekas, D. P., and Tsitsiklis, J. N., 2008. Introduction to Probability, 2nd Edition, Athena Scientific, Belmont, MA.

[BeY09] Bertsekas, D. P., and Yu, H., 2009. "Projected Equation Methods for Approximate Solution of Large Linear Systems," J. of Computational and Applied Mathematics, Vol. 227, pp. 27-50.

[BeY16] Bertsekas, D. P., and Yu, H., 2016. "Stochastic Shortest Path Problems Under Weak Conditions," Lab. for Information and Decision Systems Report LIDS-2909, MIT.

[Bel57] Bellman, R., 1957. Dynamic Programming, Princeton University Press, Princeton, N. J.

[Ber70] Bertsekas, D. P., 1970. "On the Separation Theorem for Linear Systems, Quadratic Criteria, and Correlated Noise," Unpublished Report, Electronic Systems Lab., Massachusetts Institute of Technology.

[Ber71] Bertsekas, D. P., 1971. "Control of Uncertain Systems With a Set-Membership Description of the Uncertainty," Ph.D. Dissertation, Massachusetts Institute of Technology, Cambridge, MA (available from the author's website).

[Ber72a] Bertsekas, D. P., 1972. "Infinite Time Reachability of State Space Regions by Using Feedback Control," IEEE Trans. Automatic Control, Vol. AC-17, pp. 604-613.

[Ber72b] Bertsekas, D. P., 1972. "On the Solution of Some Minimax Control Problems," Proc. 1972 IEEE Decision and Control Conf., New Orleans, LA.

[Ber74] Bertsekas, D. P., 1974. "Necessary and Sufficient Conditions for Existence of an Optimal Portfolio," J. Econ. Theory, Vol. 8, pp. 235-247.

[Ber75] Bertsekas, D. P., 1975. "Convergence of Discretization Procedures in Dynamic Programming," IEEE Trans. Automatic Control, Vol. AC-20, pp. 415-419.

[Ber76] Bertsekas, D. P., 1976. Dynamic Programming and Stochastic Control, Academic Press, N. Y.

[Ber82] Bertsekas, D. P., 1982. "Distributed Dynamic Programming," IEEE Trans. Automatic Control, Vol. AC-27, pp. 610-616.

[Ber91] Bertsekas, D. P., 1991. Linear Network Optimization: Algorithms and Codes, M.I.T. Press, Cambridge, MA.

[Ber93] Bertsekas, D. P., 1993. "A Simple and Fast Label Correcting Algorithm for Shortest Paths," Networks, Vol. 23, pp. 703-709.

[Ber97] Bertsekas, D. P., 1997. "Differential Training of Rollout Policies," Proc. of the 35th Allerton Conference on Communication, Control, and Computing, Allerton Park, Ill.

[Ber98a] Bertsekas, D. P., 1998. Network Optimization: Continuous and Discrete Models, Athena Scientific, Belmont, MA.

[Ber98b] Bertsekas, D. P., 1998. "A New Value Iteration Method for the Average Cost Dynamic Programmming Problem," SIAM J. on Control and Optimization, Vol. 36, pp. 742-759.

[Ber05a] Bertsekas, D. P., 2005. "Dynamic Programming and Suboptimal Control: A Survey from ADP to MPC," European J. of Control, Vol. 11, pp. 310-334.

[Ber05b] Bertsekas, D. P., 2005. "Rollout Algorithms for Constrained Dynamic Programming," LIDS Report 2646, MIT.

[Ber07] Bertsekas, D. P., 2007. "Separable Dynamic Programming and Approximate Decomposition Methods," IEEE Trans. on Aut. Control, Vol. 52, pp. 911-916.

[Ber09] Bertsekas, D. P., 2009. Convex Optimization Theory, Athena Scientific, Belmont, MA.

[Ber11] Bertsekas, D. P., 2011. "Approximate Policy Iteration: A Survey and Some New Methods," J. of Control Theory and Applications, Vol. 9, pp. 310-335.

[Ber13a] Bertsekas, D. P., 2013. Abstract Dynamic Programming, Athena Scientific, Belmont, MA; a 2nd edition appeared in 2018, and can be downloaded from the author's website.

[Ber13b] Bertsekas, D. P., 2013. "Rollout Algorithms for Discrete Optimization: A Survey," Handbook of Combinatorial Optimization, Springer.

[Ber14] Bertsekas, D. P., 2014. "Robust Shortest Path Planning and Semicontractive Dynamic Programming," Lab. for Information and Decision Systems Report LIDS-P-2915, MIT; arXiv preprint arXiv:1608.01670; Naval Research Logistics, Vol. 66, pp. 15-37.

[Ber15a] Bertsekas, D. P., 2015. Convex Optimization Algorithms, Athena Scientific, Belmont, MA.

[Ber15b] Bertsekas, D. P., 2015. "Regular Policies in Abstract Dynamic Programming," Lab. for Information and Decision Systems Report LIDS-P-3173, MIT; arXiv preprint arXiv:1609.03115.

[Ber15c] Bertsekas, D. P., 2015. "Value and Policy Iteration in Deterministic Optimal Control and Adaptive Dynamic Programming," arXiv preprint arXiv:1507.01026; to appear in IEEE Transactions on Neural Networks and Learning Systems.

[Ber16a] Bertsekas, D. P., 2016. Nonlinear Programming, 3rd Edition, Athena Scientific, Belmont, MA.

[Ber16b] Bertsekas, D. P., 2016. "Affine Monotonic and Risk-Sensitive Models in Dynamic Programming," Lab. for Information and Decision Systems Report LIDS-3204, MIT; arXiv preprint arXiv:1608.01393.

[Ber19a] Bertsekas, D. P., 2019. Reinforcement Learning and Optimal Control, Athena Scientific, Belmont, MA.

[Ber19b] Bertsekas, D. P., 2019. "Multiagent Rollout Algorithms and Reinforcement Learning," arXiv preprint arXiv:1910.00120.

[Ber19c] Bertsekas, D. P., 2019. "Constrained Multiagent Rollout and Multidimensional Assignment with the Auction Algorithm," arXiv preprint, arxiv.org/abs/2002.07407.

[Ber20a] Bertsekas, D. P., 2020. Rollout, Policy Iteration, and Distributed Reinforcement Learning, Athena Scientific, Belmont, MA.

[Ber20b] Bertsekas, D. P., 2020. "Multiagent Value Iteration Algorithms in Dynamic Programming and Reinforcement Learning," arXiv preprint, arxiv.org/abs/2005.01627.

[BiL97] Birge, J. R., and Louveaux, 1997. Introduction to Stochastic Programming, Springer, New York, N. Y.

[Bis95] Bishop, C. M, 1995. Neural Networks for Pattern Recognition, Oxford University Press, N. Y.

[BlT00] Blondel, V. D., and Tsitsiklis, J. N., 2000. “A Survey of Computational Complexity Results in Systems and Control,” Automatica, Vol. 36, pp. 1249-1274.

[Bla99] Blanchini, F., 1999. “Set Invariance in Control – A Survey,” Automatica, Vol. 35, pp. 1747-1768.

[BoV79] Borkar, V., and Varaiya, P. P., 1979. “Adaptive Control of Markov Chains, I: Finite Parameter Set,” IEEE Trans. Automatic Control, Vol. AC-24, pp. 953-958.

[BoV04] Boyd, S., and Vandenbergue, L., 2004. Convex Optimization, Cambridge Univ. Press, Cambridge, U.K.

[CFH05] Chang, H. S., Hu, J., Fu, M. C., and Marcus, S. I., 2005. “An Adaptive Sampling Algorithm for Solving Markov Decision Processes,” Operations Research, Vol. 53, pp. 126-139.

[CFH13] Chang, H. S., Hu, J., Fu, M. C., and Marcus, S. I., 2013. Simulation-Based Algorithms for Markov Decision Processes, (2nd Ed.), Springer, N. Y.

[CFH16] Chang, H. S., Hu, J., Fu, M. C., and Marcus, S. I., 2016. “Google DeepMind’s AlphaGo,” ORMS Today, Informs, Vol. 43.

[CGC04] Chang, H. S., Givan, R. L., and Chong, E. K. P., 2004. “Parallel Rollout for Online Solution of Partially Observable Markov Decision Processes,” Discrete Event Dynamic Systems, Vol. 14, pp. 309-341.

[CHH02] Campbell, M., Hoane, A. J. and Hsu, F. H., 2002. “Deep Blue,” Artificial Intelligence, Vol. 134, pp. 57-83.

[CaB04] Camacho, E. F., and Bordons, C., 2004. Model Predictive Control, 2nd Edition, Springer, New York, N. Y.

[Cao07] Cao, X. R., 2007. Stochastic Learning and Optimization: A Sensitivity-Based Approach, Springer, N. Y.

[ChT89] Chow, C.-S., and Tsitsiklis, J. N., 1989. “The Complexity of Dynamic Programming,” J. of Complexity, Vol. 5, pp. 466-488.

[ChT91] Chow, C.-S., and Tsitsiklis, J. N., 1991. “An Optimal One–Way Multigrid Algorithm for Discrete–Time Stochastic Control,” IEEE Trans. on Automatic Control, Vol. AC-36, 1991, pp. 898-914.

[ChV12] Chacon, A., and Vladimirsky, A., 2012. “Fast Two-Scale Methods for Eikonal Equations,” SIAM J. on Scientific Computing, Vol. 34, pp. A547-A578.

[ChV13] Chacon, A., and Vladimirsky, A., 2013. “A Parallel Heap-Cell Method for Eikonal Equations,” arXiv preprint arXiv:1306.4743.

[ChV15] Chacon, A., and Vladimirsky, A., 2015. “A Parallel Two-Scale Method for Eikonal Equations,” SIAM J. on Scientific Computing, Vol. 37, pp. A156-A180.

[Che72] Chernoff, H., 1972. “Sequential Analysis and Optimal Design,” Regional Conference Series in Applied Mathematics, SIAM, Philadelphia, PA.

[CoL55] Coddington, E. A., and Levinson, N., 1955. Theory of Ordinary Differential Equations, McGraw-Hill, N. Y.

[Cou06] Coulom, R., 2006. “Efficient Selectivity and Backup Operators in Monte-Carlo Tree Search,” International Conference on Computers and Games, Springer, pp. 72-83.

[Cyb89] Cybenko, 1989. “Approximation by Superpositions of a Sigmoidal Function,” Math. of Control, Signals, and Systems, Vol. 2, pp. 303-314.

[DeG70] DeGroot, M. H., 1970. Optimal Statistical Decisions, McGraw-Hill, N. Y.

[DeP84] Deo, N., and Pang, C., 1984. "Shortest Path Problems: Taxonomy and Annotation," Networks, Vol. 14, pp. 275-323.

[DeR79] Denardo, E. V., and Rothblum, U. G., 1979. "Optimal Stopping, Exponential Utility, and Linear Programming," Math. Programming, Vol. 16, pp. 228-244.

[Del89] Deller, J. R., 1989. "Set Membership Identification in Digital Signal Processing," IEEE ASSP Magazine, Oct., pp. 4-20.

[DoS80] Doshi, B., and Shreve, S., 1980. "Strong Consistency of a Modified Maximum Likelihood Estimator for Controlled Markov Chains," J. of Applied Probability, Vol. 17, pp. 726-734.

[Dre65] Dreyfus, S. D., 1965. Dynamic Programming and the Calculus of Variations, Academic Press, N. Y.

[Dre69] Dreyfus, S. D., 1969. "An Appraisal of Some Shortest-Path Algorithms," Operations Research, Vol. 17, pp. 395-412.

[Eck68] Eckles, J. E., 1968. "Optimum Maintenance with Incomplete Information," Operations Research, Vol. 16, pp. 1058-1067.

[Elm78] Elmaghraby, S. E., 1978. Activity Networks: Project Planning and Control by Network Models, Wiley-Interscience, N. Y.

[FGL13] Festa, P., Guerriero, F., Lagana, D. and Musmanno, R., 2013. "Solving the Shortest Path Tour Problem," European J. of Operational Research, Vol. 230, pp. 464-474.

[FYG06] Fern, A., Yoon, S. and Givan, R., 2006. "Approximate Policy Iteration with a Policy Language Bias: Solving Relational Markov Decision Processes," J. of Artificial Intelligence Research, Vol. 25, pp. 75-118.

[Fal87] Falcone, M., 1987. "A Numerical Approach to the Infinite Horizon Problem of Deterministic Control Theory," Appl. Math. Opt., Vol. 15, pp. 1-13.

[FeM94] Fernandez-Gaucherand, E., and Marcus, S. I., 1994. "Risk Sensitive Optimal Control of Hidden Markov Models," Proc. 33rd IEEE Conf. Dec. Control, Lake Buena Vista, Fla.

[FeV02] Ferris, M. C., and Voelker, M. M., 2002. "Neuro-Dynamic Programming for Radiation Treatment Planning," Numerical Analysis Group Research Report NA-02/06, Oxford University Computing Laboratory, Oxford University.

[FeV04] Ferris, M. C., and Voelker, M. M., 2004. "Fractionation in Radiation Treatment Planning," Mathematical Programming B, Vol. 102, pp. 387-413.

[Fei16] Feinberg, E. A, 2016. "Optimality Conditions for Inventory Control," INFORMS Tutorials in Operations Research, Online, pp. 14-45.

[Fel68] Feller, W., 1968. An Introduction to Probability Theory and its Applications, Wiley, N. Y.

[Fis70] Fishburn, P. C., 1970. Utility Theory for Decision Making, Wiley, N. Y.

[For56] Ford, L. R., Jr., 1956. "Network Flow Theory," Report P-923, The Rand Corporation, Santa Monica, CA.

[For73] Forney, G. D., 1973. "The Viterbi Algorithm," Proc. IEEE, Vol. 61, pp. 268-278.

[Fox71] Fox, B. L., 1971. "Finite State Approximations to Denumerable State Dynamic Progams," J. Math. Anal. Appl., Vol. 34, pp. 665-670.

[Fun89] Funahashi, K., 1989. “On the Approximate Realization of Continuous Mappings by Neural Networks,” Neural Networks, Vol. 2, pp. 183-192.

[GBC16] Goodfellow, I., Bengio, J., and Courville, A., Deep Learning, MIT Press, Cambridge, MA.

[GGS13] Gabillon, V., Ghavamzadeh, M., and Scherrer, B., 2013. “Approximate Dynamic Programming Finally Performs Well in the Game of Tetris,” in Advances in Neural Information Processing Systems, pp. 1754-1762.

[GTO15] Goodson, J. C., Thomas, B. W., and Ohlmann, J. W., 2015. “Restocking-Based Rollout Policies for the Vehicle Routing Problem with Stochastic Demand and Duration Limits,” Transportation Science, Vol. 50, pp. 591-607.

[GaP88] Gallo, G., and Pallottino, S., 1988. “Shortest Path Algorithms,” Annals of Operations Research, Vol. 7, pp. 3-79.

[Gal13] Gallager, R. G., 2013. Stochastic Processes, Cambridge Univ. Press.

[GlS71] Glover, J., and Schweppe, F., 1971. “Control of Linear Dynamic Systems with Set Constrained Disturbances,” IEEE Trans. on Automatic Control, Vol. 16, pp. 411-423.

[GoR85] Gonzalez, R., and Rofman, E., 1985. “On Deterministic Control Problems: An Approximation Procedure for the Optimal Cost, Parts I, II,” SIAM J. Control Optimization, Vol. 23, pp. 242-285.

[GoS84] Goodwin, G. C., and Sin, K. S. S., 1984. Adaptive Filtering, Prediction, and Control, Prentice-Hall, Englewood Cliffs, N. J.

[Gos15] Gosavi, A., 2015. Simulation-Based Optimization: Parametric Optimization Techniques and Reinforcement Learning, 2nd Edition, Springer, N. Y.

[GrA66] Groen, G. J., and Atkinson, R. C., 1966. “Models for Optimizing the Learning Process,” Psychol. Bull., Vol. 66, pp. 309-320.

[Gre11] Grewal, M. S., 2011. Kalman Filtering, Springer, Berlin.

[GuF63] Gunckel, T. L., and Franklin, G. R., 1963. “A General Solution for Linear Sampled-Data Control,” Trans. ASME Ser. D. J. Basic Engrg., Vol. 85, pp. 197-20l.

[GuM01] Guerriero, F., and Musmanno, R., 2001. “Label Correcting Methods to Solve Multicriteria Shortest Path Problems,” J. Optimization Theory Appl., Vol. 111, pp. 589-613.

[GuM03] Guerriero, F., and Mancini, M., 2003. “A Cooperative Parallel Rollout Algorithm for the Sequential Ordering Problem,” Parallel Computing, Vol. 29, pp. 663-677.

[HJG16] Huang, Q., Jia, Q. S., and Guan, X., 2016. “Robust Scheduling of EV Charging Load with Uncertain Wind Power Integration,” IEEE Transactions on Smart Grid.

[HMS55] Holt, C. C., Modigliani, F., and Simon, H. A., 1955. “A Linear Decision Rule for Production and Employment Scheduling,” Management Sci., Vol. 2, pp. 1-30.

[HNR68] Hart, P. E., Nilsson, N. J. and Raphael, B., 1968. “A Formal Basis for the Heuristic Determination of Minimum Cost Paths,” IEEE Trans. on Systems Science and Cybernetics, Vol. 4, pp. 100-107.

[HNR72] Hart, P. E., Nilsson, N. J. and Raphael, B., 1972. “Correction to a Formal Basis for the Heuristic Determination of Minimum Cost Paths,” ACM SIGART Bulletin, Vol. 37, pp. 28-29.

[HPC96] Helmsen, J., Puckett, E. G., Colella, P., and Dorr, M., 1996. “Two New Methods for Simulating Photolithography Development," SPIE's 1996 International Symposium on Microlithography, pp. 253-261.

[HSW89] Hornick, K., Stinchcombe, M., and White, H., 1989. "Multilayer Feedforward Networks are Universal Approximators," Neural Networks, Vol. 2, pp. 359-159.

[HaA15] Hansen, E. A. and Abdoulahi, I., 2015. "Efficient Bounds in Heuristic Search Algorithms for Stochastic Shortest Path Problems," in AAAI, pp. 3283-3290.

[HaL82] Hajek, B., and van Loon, T., 1982. "Decentralized Dynamic Control of a Multiaccess Broadcast Channel," IEEE Trans. Automatic Control, Vol. AC-27, pp. 559-569.

[Hak70] Hakansson, N. H., 1970. "Optimal Investment and Consumption Strategies under Risk for a Class of Utility Functions," Econometrica, Vol. 38, pp. 587-607.

[Hak71] Hakansson, N. H., 1971. "On Myopic Portfolio Policies, With and Without Serial Correlation of Yields," The J. of Business of the University of Chicago, Vol. 44, pp. 324-334.

[Han80] Hansen, P., 1980. "Bicriterion Path Problems," in Multiple-Criteria Decision Making: Theory and Applications, Edited by G. Fandel and T. Gal, Springer Verlag, Heidelberg, Germany, pp. 109-127.

[Han06] Hansen, N., 2006. "The CMA Evolution Strategy: A Comparing Review," in Towards a New Evolutionary Computation, Springer Berlin Heidelberg, pp. 75-102.

[Han17] Hansen, E. A., 2017. "Error Bounds for Stochastic Shortest Path Problems," Math. of Operations Research, forthcoming.

[Hay09] Haykin, S., 2009. Neural Networks and Learning Machines, (3rd Edition), Prentice-Hall, Englewood-Cliffs, N. J.

[Her89] Hernandez-Lerma, O., 1989. Adaptive Markov Control Processes, Springer, N. Y.

[Hes66] Hestenes, M. R., 1966. Calculus of Variations and Optimal Control Theory, Wiley, N. Y.

[HoK71] Hoffman, K., and Kunze, R., 1971. Linear Algebra, 2nd ed., Prentice-Hall, Englewood Cliffs, N. J.

[IEE71] IEEE Trans. Automatic Control, 1971. Special Issue on Linear-Quadratic Gaussian Problem, Vol. AC-16.

[IoS96] Ioannou, P. A., and Sun, J., 1996. Robust Adaptive Control, Prentice-Hall, Englewood Cliffs, N. J.

[JBE94] James, M. R., Baras, J. S., and Elliott, R. J., 1994. "Risk-Sensitive Control and Dynamic Games for Partially Observed Discrete-Time Nonlinear Systems," IEEE Trans. on Automatic Control, Vol. AC-39, pp. 780-792.

[Jac73] Jacobson, D. H., 1973. "Optimal Stochastic Linear Systems With Exponential Performance Criteria and their Relation to Deterministic Differential Games," IEEE Trans. Automatic Control, Vol. AC-18, pp. 124-131.

[Jaf84] Jaffe, J. M., 1984. "Algorithms for Finding Paths with Multiple Constraints," Networks, Vol. 14, pp. 95-116.

[Jew63] Jewell, W., 1963. "Markov Renewal Programming I and II," Operations Research, Vol. 2, pp. 938-971.

[JoT61] Joseph, P. D., and Tou, J. T., 1961. "On Linear Control Theory," AIEE Trans., Vol. 80 (II), pp. 193-196.

[Jon90] Jones, L. K., 1990. "Constructive Approximations for Neural Networks by Sigmoidal Functions," Proceedings of the IEEE, Vol. 78, pp. 1586-1589.

[KGB82] Kimemia, J., Gershwin, S. B., and Bertsekas, D. P., 1982. "Computation of Production Control Policies by a Dynamic Programming Technique," in Analysis and Optimization of Systems, A. Bensoussan and J. L. Lions (eds.), Springer, N. Y., pp. 243-269.

[KKK95] Krstic, M., Kanellakopoulos, I., Kokotovic, P., 1995. Nonlinear and Adaptive Control Design, J. Wiley, N. Y.

[KLB92] Kosut, R. L., Lau, M. K., and Boyd, S. P., 1992. "Set-Membership Identification of Systems with Parametric and Nonparametric Uncertainty," IEEE Trans. on Automatic Control, Vol. AC-37, pp. 929-941.

[KLC98] Kaelbling, L.P., Littman, M. L., and Cassandra, A. R., 1998. "Planning and Acting in Partially Observable Stochastic Domains," Artificial Intelligence, Vol. 101, pp. 99-134.

[KaD66] Karush, W., and Dear, E. E., 1966. "Optimal Stimulus Presentation Strategy for a Stimulus Sampling Model of Learning," J. Math. Psychology, Vol. 3, pp. 15-47.

[KaK58] Kalman, R. E., and Koepcke, R. W., 1958. "Optimal Synthesis of Linear Sampling Control Systems Using Generalized Performance Indexes," Trans. ASME, Vol. 80, pp. 1820-1826.

[KaW94] Kall, P., and Wallace, S. W., 1994. Stochastic Programming, Wiley, Chichester, UK.

[Kal60] Kalman, R. E., 1960. "A New Approach to Linear Filtering and Prediction Problems," Trans. ASME Ser. D. J. Basic Engrg., Vol. 82, pp. 35-45.

[KeG88] Keerthi, S. S., and Gilbert, E. G., 1988. "Optimal, Infinite Horizon Feedback Laws for a General Class of Constrained Discrete Time Systems: Stability and Moving-Horizon Approximations," J. Optimization Theory Appl., Vo. 57, pp. 265-293.

[Kim82] Kimemia, J., 1982. "Hierarchical Control of Production in Flexible Manufacturing Systems," Ph.D. Thesis, Dep. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology.

[KoC15] Kouvaritakis, B., and Cannon, M., 2015. Model Predictive Control: Classical, Robust and Stochastic, Springer, N. Y.

[KoS06] Kocsis, L., and Szepesvari, C., 2006. "Bandit Based Monte-Carlo Planning," Proc. of 17th European Conference on Machine Learning, Berlin, pp. 282-293.

[KuA77] Ku, R., and Athans, M., 1977. "Further Results on the Uncertainty Threshold Principle," IEEE Trans. on Automatic Control, Vol. AC-22, pp. 866-868.

[KuD92] Kushner, H. J., and Dupuis, P. G., 1992. Numerical Methods for Stochastic Control Problems in Continuous Time, Springer, N. Y.

[KuL82] Kumar, P. R., and Lin, W., 1982. "Optimal Adaptive Controllers for Unknown Markov Chains," IEEE Trans. Automatic Control, Vol. AC-27, pp. 765-774.

[KuV86] Kumar, P. R., and Varaiya, P. P., 1986. Stochastic Systems: Estimation, Identification, and Adaptive Control, Prentice-Hall, Englewood Cliffs, N. J.

[KuV97] Kurzhanski, A., and Valyi, I., 1997. Ellipsoidal Calculus for Estimation and Control, Birkhauser, Boston, MA.

[Kum83] Kumar, P. R., 1983. "Optimal Adaptive Control of Linear-Quadratic-Gaussian Systems," SIAM J. on Control and Optimization, Vol. 21, pp. 163-178.

[Kum85] Kumar, P. R., 1985. "A Survey of Some Results in Stochastic Adaptive Control," SIAM J. on Control and Optimization, Vol. 23, pp. 329-380.

[LGW16] Lan, Y., Guan, X., and Wu, J., 2016. "Rollout Strategies for Real-Time Multi-Energy Scheduling in Microgrid with Storage System," IET Generation, Transmission and Distribution, Vol. 10, pp. 688-696.

[LLL08] Lewis, F. L., Liu, D., and Lendaris, G. G., 2008. Special Issue on Adaptive Dynamic Programming and Reinforcement Learning in Feedback Control, IEEE Trans. on Systems, Man, and Cybernetics, Part B, Vol. 38, No. 4.

[LLP93] Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S., 1993. "Multilayer Feedforward Networks with a Nonpolynomial Activation Function can Approximate any Function," Neural Networks, Vol. 6, pp. 861-867.

[LWW17] Liu, D., Wei, Q., Wang, D., and Yang, X., 2017. Adaptive Dynamic Programming with Applications in Optimal Control, Springer, Berlin.

[LaW17] Lam, R., and Willcox, K., 2017. "Lookahead Bayesian Optimization with Inequality Constraints," in Advances in Neural Information Processing Systems, pp. 1890-1900.

[Las85] Lasserre, J. B., 1985. "A Mixed Forward-Backward Dynamic Programming Method Using Parallel Computation," J. Optimization Theory Appl., Vol. 45, pp. 165-168.

[LeL13] Lewis, F. L., and Liu, D., (Eds), 2013. Reinforcement Learning and Approximate Dynamic Programming for Feedback Control, Wiley, Hoboken, N. J.

[Lev84] Levy, D., 1984. The Chess Computer Handbook, B. T. Batsford Ltd., London.

[LiR71] Lippman, S. A., and Ross, S. M., 1971. "The Streetwalker's Dilemma: A Job-Shop Model," SIAM J. of Appl. Math., Vol. 20, pp. 336-342.

[LiR06] Lincoln, B., and Rantzer, A., 2006. "Relaxing Dynamic Programming," IEEE Trans. Automatic Control, Vol. 51, pp. 1249-1260.

[LiW15] Li, H. and Womer, N. K., 2015. "Solving Stochastic Resource-Constrained Project Scheduling Problems by Closed-Loop Approximate Dynamic Programming," European J. of Operational Research, Vol. 246, pp. 20-33.

[LjS83] Ljung, L., and Soderstrom, T., 1983. Theory and Practice of Recursive Identification, MIT Press, Cambridge, MA.

[Lov91a] Lovejoy, W. S., 1991. "Computationally Feasible Bounds for Partially Observed Markov Decision Processes," Operations Research, Vol. 39, pp. 162175.

[Lov91b] Lovejoy, W. S., 1991. "A Survey of Algorithmic Methods for Partially Observed Markov Decision Processes," Annals of Operations Research, Vol. 18, pp. 47-66.

[Lue69] Luenberger, D. G., 1969. Optimization by Vector Space Methods, Wiley, N. Y.

[Lue84] Luenberger, D. G., 1984. Linear and Nonlinear Programming, Addison-Wesley, Reading, MA.

[MHK98] Meuleau, N., Hauskrecht, M., Kim, K.-E., Peshkin, L., Kaelbling, L. K., and Dean, T., 1998. "Solving Very Large Weakly Coupled Markov Decision Processes," Proc. of the Fifteenth National Conference on Artificial Intelligence, Madison, WI, pp. 165-172.

[MMB02] McGovern, A., Moss, E., and Barto, A., 2002. "Building a Basic Building Block Scheduler Using Reinforcement Learning and Rollouts," Machine Learning, Vol. 49, pp. 141-160.

[MPP04] Meloni, C., Pacciarelli, D., and Pranzo, M., 2004. "A Rollout Metaheuristic for Job Shop Scheduling Problems," Annals of Operations Research, Vol. 131, pp. 215-235.

[MRR00] Mayne, D., Rawlings, J. B., Rao, C. V., and Scokaert, P. O. M., 2000. "Constrained Model Predictive Control: Stability and Optimality," Automatica, Vol. 36, pp. 789-814.

[MVZ95] Mulvey, J. M., Vanderbei, R. J., and Zenios, S. A., 1995. "Robust Optimization of Large-Scale Systems," Operations Research, Vol. 43, pp. 264-281.

[MaJ15] Mastin, A., and Jaillet, P., 2015. "Average-Case Performance of Rollout Algorithms for Knapsack Problems," J. of Optimization Theory and Applications, Vol. 165, pp. 964-984.

[Mac02] Maciejowski, J. M., 2002. Predictive Control with Constraints, Addison-Wesley, Reading, MA.

[Mar84] Martins, E. Q. V., 1984. "On a Multicriteria Shortest Path Problem," European J. of Operational Research, Vol. 16, pp. 236-245.

[May14] Mayne, D. Q., 2014. "Model Predictive Control: Recent Developments and Future Promise," Automatica, Vol. 50, pp. 2967-2986.

[McQ66] MacQueen, J., 1966. "A Modified Dynamic Programming Method for Markovian Decision Problems," J. Math. Anal. Appl., Vol. 14, pp. 38-43.

[Mik79] Mikhailov, V. A., 1979. Methods of Random Multiple Access, Candidate Engineering Thesis, Moscow Institute of Physics and Technology, Moscow.

[MoL99] Morari, M., and Lee, J. H., 1999. "Model Predictive Control: Past, Present, and Future," Computers and Chemical Engineering, Vol. 23, pp. 667-682.

[Mos68] Mossin, J., 1968. "Optimal Multi-Period Portfolio Policies," J. Business, Vol. 41, pp. 215-229.

[MuS08] Munos, R., and Szepesvari, C, 2008. "Finite-Time Bounds for Fitted Value Iteration," J. of Machine Learning Research, Vol. 1, pp. 815-857.

[Mun14] Munos, R., 2014. "From Bandits to Monte-Carlo Tree Search: The Optimistic Principle Applied to Optimization and Planning," Foundations and Trends in Machine Learning, Vol. 7, pp. 1-129.

[NeW88] Nemhauser, G. L., and Wolsey, L. A., 1988. Integer and Combinatorial Optimization, Wiley, N. Y.

[New75] Newborn, M., 1975. Computer Chess, Academic Press, N. Y.

[Nic66] Nicholson, T., 1966. "Finding the Shortest Route Between Two Points in a Network," The Computer Journal, Vol. 9, pp. 275-280.

[Nil71] Nilsson, N. J., 1971. Problem-Solving Methods in Artificial Intelligence, McGraw-Hill, N. Y.

[Nil80] Nilsson, N. J., 1980. Principles of Artificial Intelligence, Morgan-Kaufmann, San Mateo, Ca.

[OBP16] Osband, I., Blundell, C., Pritzel, A., and Van Roy, B., 2016. "Deep Exploration Via Bootstrapped DQN," Advances in Neural Information Processing Systems, Vol. 29.

[OVW16] Osband, I., Van Roy, B., and Wen, Z., 2016. "Generalization and Exploration Via Randomized Value Functions," Proc. of the 33rd International Conference on Machine Learning, pp. 2377-2386.

[Osb16] Osband, I., 2016. Deep Exploration via Randomized Value Functions, Ph.D. Dissertation, Stanford University.

[PBG65] Pontryagin, L. S., Boltyanski, V., Gamkrelidze, R., and Mishchenko, E., 1965. The Mathematical Theory of Optimal Processes, Interscience Publishers, Inc., N. Y.

[PBT98] Polymenakos, L. C., Bertsekas, D. P., and Tsitsiklis, J. N., 1998. “Efficient Algorithms for Continuous-Space Shortest Path Problems,” IEEE Trans. on Automatic Control, Vol. 43, pp. 278-283.

[PaT87] Papadimitriou, C. H., and Tsitsiklis, J. N., 1987. “The Complexity of Markov Decision Processes,” Math. Operations Res., Vol. 12, pp. 441-450.

[Pap74] Pape, V., 1974. “Implementation and Efficiency of Moore Algorithms for the Shortest Path Problem,” Math. Progr., Vol. 7, pp. 212-222.

[Pat01] Patek, S. D., 2001. “On Terminating Markov Decision Processes with a Risk Averse Objective Function,” Automatica, Vol. 37, pp. 1379-1386.

[Pea84] Pearl, J., 1984. Heuristics, Addison-Wesley, Reading, MA.

[Pic90] Picone, J., 1990. “Continuous Speech Recognition Using Hidden Markov Models,” IEEE ASSP Magazine, July Issue, pp. 26-41.

[Pin95] Pinedo, M., 1995. Scheduling: Theory, Algorithms, and Systems, Prentice-Hall, Englewood Cliffs, N. J.

[Pos14] Post, H. N., 2014. The Shortest Path Problem on Real Road Networks: Theory, Algorithms and Computations, TU Delft, Delft University of Technology.

[Pow07] Powell, W. B., 2007. Approximate Dynamic Programming: Solving the Curses of Dimensionality, J. Wiley and Sons, Hoboken, N. J; a 2nd edition appeared in 2011.

[PrS94] Proakis, J. G., and Salehi, M., 1994. Communication Systems Engineering, Prentice-Hall, Englewood Cliffs, N. J.

[Pra64] Pratt, J. W., 1964. “Risk Aversion in the Small and in the Large,” Econometrica, Vol. 32, pp. 300-307.

[Pre95] Prekopa, A., 1995. Stochastic Programming, Kluwer, Boston.

[RMD17] Rawlings, J. B., Mayne, D. Q., and Diehl, M. M., 2017. Model Predictive Control: Theory, Computation, and Design, 2nd Ed., Nob Hill Publishing.

[RSS12] Runarsson, T. P., Schoenauer, M., and Sebag, M., 2012. “Pilot, Rollout and Monte Carlo Tree Search Methods for Job Shop Scheduling,” in Learning and Intelligent Optimization, Springer, Berlin, pp. 160-174.

[Rab89] Rabiner, L. R., 1989. “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition,” Proc. of the IEEE, Vol. 77, pp. 257-286.

[Roc70] Rockafellar, R. T., 1970. Convex Analysis, Princeton University Press, Princeton, N. J.

[Ros70] Ross, S. M., 1970. Applied Probability Models with Optimization Applications, Holden-Day, San Francisco, CA.

[Ros83] Ross, S. M., 1983. Introduction to Stochastic Dynamic Programming, Academic Press, N. Y.

[Ros85] Ross, S. M., 1985. Probability Models, Academic Press, Orlando, Fla.

[Ros12] Ross, S. M., 2012. Simulation, 5th Edition, Academic Press, Orlando, Fla.

[RuK04] Rubinstein, R. Y., and Kroese, D. P., 2004. The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Springer, N. Y.

[RuK17] Rubinstein, R. Y., and Kroese, D. P., 2017. Simulation and the Monte Carlo Method, 3rd Edition, J. Wiley, N. Y.

[Rud76] Rudin, W., 1976. Real Analysis, 3rd Edition, McGraw-Hill, N. Y.

[Rus97] Rust, J., 1997. "Using Randomization to Break the Curse of Dimensionality," Econometrica, Vol. 65, pp. 487-516.

[SBB89] Sastry, S., Bodson, M., and Bartram, J. F., 1989. Adaptive Control: Stability, Convergence, and Robustness, Prentice-Hall, Englewood Cliffs, N. J.

[SBP04] Si, J., Barto, A., Powell, W., and Wunsch, D., (Eds.) 2004. Learning and Approximate Dynamic Programming, IEEE Press, N. Y.

[SGC02] Savagaonkar, U., Givan, R., and Chong, E. K. P., 2002. "Sampling Techniques for Zero-Sum, Discounted Markov Games," in Proc. 40th Allerton Conference on Communication, Control and Computing, Monticello, Ill.

[SGG15] Scherrer, B., Ghavamzadeh, M., Gabillon, V., Lesner, B., and Geist, M., 2015. "Approximate Modified Policy Iteration and its Application to the Game of Tetris," J. of Machine Learning Research, Vol. 16, pp. 1629-1676.

[SHB15] Simroth, A., Holfeld, D., and Brunsch, R., 2015. "Job Shop Production Planning under Uncertainty: A Monte Carlo Rollout Approach," Proc. of the International Scientific and Practical Conference, Vol. 3, pp. 175-179.

[SHM16] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., and Dieleman, S., 2016. "Mastering the Game of Go with Deep Neural Networks and Tree Search," Nature, Vol. 529, pp. 484-489.

[SHS17] Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., and Lillicrap, T., 2017. "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm," arXiv preprint arXiv:1712.01815.

[SLA12] Snoek, J., Larochelle, H., and Adams, R. P., 2012. "Practical Bayesian Optimization of Machine Learning Algorithms," in Advances in Neural Information Processing Systems, pp. 2951-2959.

[SLJ13] Sun, B., Luh, P. B., Jia, Q. S., Jiang, Z., Wang, F., and Song, C., 2013. "Building Energy Management: Integrated Control of Active and Passive Heating, Cooling, Lighting, Shading, and Ventilation Systems," IEEE Trans. on Automation Science and Engineering, Vol. 10, pp. 588-602.

[SNC18] Sarkale, Y., Nozhati, S., Chong, E. K., Ellingwood, B. R., and Mahmoud, H., 2018. "Solving Markov Decision Processes for Network-Level Post-Hazard Recovery via Simulation Optimization and Rollout," in 2018 IEEE 14th International Conference on Automation Science and Engineering, pp. 906-912.

[SZL08] Sun, T., Zhao, Q., Lun, P., and Tomastik, R., 2008. "Optimization of Joint Replacement Policies for Multipart Systems by a Rollout Framework," IEEE Transactions on Automation Science and Engineering, Vol. 5, pp. 609-619.

[SaB11] Sastry, S., and Bodson, M., 2011. Adaptive Control: Stability, Convergence and Robustness, Courier Corporation.

[Sam69] Samuelson, P. A., 1969. "Lifetime Portfolio Selection by Dynamic Stochastic Programming," Review of Economics and Statistics, Vol. 51, pp. 239-246.

[Sar87] Sargent, T. J., 1987. Dynamic Macroeconomic Theory, Harvard Univ. Press, Cambridge, MA.

[Sca60] Scarf, H., 1960. "The Optimality of (s, S) Policies for the Dynamic Inventory Problem," Proceedings of the 1st Stanford Symposium on Mathematical Methods in the Social Sciences, Stanford University Press, Stanford, CA.

[Sch68] Schweppe, F. C., 1968. "Recursive State Estimation; Unknown but Bounded Errors and System Inputs," IEEE Trans. Automatic Control, Vol. AC-13.

[Sch97] Schaeffer, J., 1997. One Jump Ahead, Springer, N. Y.

[Sch13] Scherrer, B., 2013. "Performance Bounds for Lambda Policy Iteration and Application to the Game of Tetris," J. of Machine Learning Research, Vol. 14, pp. 1181-1227.

[Sec00] Secomandi, N., 2000. "Comparing Neuro-Dynamic Programming Algorithms for the Vehicle Routing Problem with Stochastic Demands," Computers and Operations Research, Vol. 27, pp. 1201-1225.

[Sec01] Secomandi, N., 2001. "A Rollout Policy for the Vehicle Routing Problem with Stochastic Demands," Operations Research, Vol. 49, pp. 796-802.

[Sec03] Secomandi, N., 2003. "Analysis of a Rollout Approach to Sequencing Problems with Stochastic Routing Applications," J. of Heuristics, Vol. 9, pp. 321-352.

[Set99a] Sethian, J. A., 1999. Level Set Methods and Fast Marching Methods Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science, Cambridge University Press, N. Y.

[Set99b] Sethian, J. A., 1999. "Fast Marching Methods," SIAM Review, Vol. 41, pp. 199-235.

[Sha50] Shannon, C., 1950. "Programming a Digital Computer for Playing Chess," Phil. Mag., Vol. 41, pp. 356-375.

[Shi64] Shiryaev, A. N., 1964. "On Markov Sufficient Statistics in Non-Additive Bayes Problems of Sequential Analysis," Theory of Probability and Applications, Vol. 9, pp. 604-618.

[Shi66] Shiryaev, A. N., 1966. "On the Theory of Decision Functions and Control by an Observation Process with Incomplete Data," Selected Translations in Math. Statistics and Probability, Vol. 6, pp. 162-188.

[Shr81] Shreve, S. E., 1981. "A Note on Optimal Switching Between Two Activities," Naval Research Logistics Quarterly, Vol. 28, pp. 185-190.

[Sim56] Simon, H. A., 1956. "Dynamic Programming Under Uncertainty with a Quadratic Criterion Function," Econometrica, Vol. 24, pp. 74-81.

[Sim06] Simon, D., 2006. Optimal State Estimation: Kalman, H-Infinity, and Nonlinear Approaches, J. Wiley, N. Y.

[Skl88] Sklar, B., 1988. Digital Communications: Fundamentals and Applications, Prentice-Hall, Englewood Cliffs, N. J.

[SlL91] Slotine, J.-J. E., and Li, W., Applied Nonlinear Control, Prentice-Hall, Englewood Cliffs, N. J.

[SmS73] Smallwood, R. D., and Sondik, E. J., 1973. "The Optimal Control of Partially Observable Markov Processes Over a Finite Horizon," Operations Res., Vol. 11, pp. 1071-1088.

[Sma71] Smallwood, R. D., 1971. "The Analysis of Economic Teaching Strategies for a Simple Learning Model," J. Math. Psychology, Vol. 8, pp. 285-301.

[Sob75] Sobel, M. J., 1975. "Ordinal Dynamic Programming," Management Science, Vol. 21, pp. 967-975.

[Son71] Sondik, E. J., 1971. "The Optimal Control of Partially Observable Markov Processes," Ph.D. Dissertation, Department of Engineering-Economic Systems, Stanford University, Stanford, CA.

[StL89] Stokey, N. L., and Lucas, R. E., 1989. Recursive Methods in Economic Dynamics, Harvard University Press, Cambridge, MA.

[StW91] Stewart, B. S., and White, C. C., 1991. "Multiobjective A^*," J. ACM, Vol. 38, pp. 775-814.

[Sti94] Stirzaker, D., 1994. Elementary Probability, Cambridge University Press, Cambridge.

[Str65] Striebel, C. T., 1965. "Sufficient Statistics in the Optimal Control of Stochastic Systems," J. Math. Anal. Appl., Vol. 12, pp. 576-592.

[Str76] Strang, G., 1976. Linear Algebra and its Applications, Academic Press, N. Y.

[SuB98] Sutton, R., and Barto, A. G., 1998. Reinforcement Learning, MIT Press, Cambridge, MA.

[SzL06] Szita, I., and Lorinz, A., 2006. "Learning Tetris Using the Noisy Cross-Entropy Method," Neural Computation, Vol. 18, pp. 2936-2941.

[Sze10] Szepesvari, C., 2010. Algorithms for Reinforcement Learning, Morgan and Claypool Publishers, San Franscisco, CA.

[TGL13] Tesauro, G., Gondek, D. C., Lenchner, J., Fan, J., and Prager, J. M., 2013. "Analysis of Watson's Strategies for Playing Jeopardy!," J. of Artificial Intelligence Research.

[TeG96] Tesauro, G., and Galperin, G. R., 1996. "On-Line Policy Improvement Using Monte Carlo Search," presented at the 1996 Neural Information Processing Systems Conference, Denver, CO; also in M. Mozer et al. (eds.), Advances in Neural Information Processing Systems 9, MIT Press (1997).

[Tes89] Tesauro, G., "Connectionist Learning of Expert Preferences by Comparison Training," in D. Touretzky (Ed.), Advances in Neural Information Processing Systems, (NIPS-88), Morgan Kaufmann, San Mateo, CA, pp. 99-106.

[Tes01] Tesauro, G., "Comparison Training of Chess Evaluation Functions," in J. Furnkranz, M. Kumbat (Eds.), Machines that Learn to Play Games, Nova Science Publishers, pp. 117-130.

[ThS09] Thiery, C., and Scherrer, B., 2009. "Improvements on Learning Tetris with Cross-Entropy," International Computer Games Association J., Vol. 32, pp. 23-33.

[The54] Theil, H., 1954. "Econometric Models and Welfare Maximization," Weltwirtsch. Arch., Vol. 72, pp. 60-83.

[TsV96] Tsitsiklis, J. N., and Van Roy, B., 1996. "Feature-Based Methods for Large-Scale Dynamic Programming," Machine Learning, Vol. 22, pp. 59-94.

[Tsi84a] Tsitsiklis, J. N., 1984. "Convexity and Characterization of Optimal Policies in a Dynamic Routing Problem," J. Optimization Theory Appl., Vol. 44, pp. 105-136.

[Tsi84b] Tsitsiklis, J. N., 1984. "Periodic Review Inventory Systems with Continuous Demand and Discrete Order Sizes," Management Sci., Vol. 30, pp. 1250-1254.

[Tsi87] Tsitsiklis, J. N., 1987. "Analysis of a Multiaccess Control Scheme," IEEE Trans. Automatic Control, Vol. AC-32, pp. 1017-1020.

[Tsi95] Tsitsiklis, J. N., 1995. "Efficient Algorithms for Globally Optimal Trajectories," IEEE Trans. Automatic Control, Vol. AC-40, pp. 1528-1538.

[TuP03] Tu, F., and Pattipati, K. R., 2003. "Rollout Strategies for Sequential Fault Diagnosis," IEEE Trans. on Systems, Man and Cybernetics, Part A, pp. 86-99.

[UGM18] Ulmer, M. W., Goodson, J. C., Mattfeld, D. C., and Hennig, M., 2018. "Offline-Online Approximate Dynamic Programming for Dynamic Vehicle Routing with Stochastic Requests," Transportation Science, Vol. 53, pp. 185-202.

[Ulm17] Ulmer, M. W., 2017. Approximate Dynamic Programming for Dynamic Vehicle Routing, Springer, Berlin.

[VVL13] Vrabie, D., Vamvoudakis, K. G., and Lewis, F. L., 2013. Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles, The Institution of Engineering and Technology, London.

[VaW89] Varaiya, P., and Wets, R. J-B., 1989. "Stochastic Dynamic Optimization Approaches and Computation," Mathematical Programming: State of the Art, M. Iri and K. Tanabe (eds.), Kluwer, Boston, pp. 309-332.

[Vei65] Veinott, A. F., Jr., 1965. "The Optimal Inventory Policy for Batch Ordering," Operations Res., Vol. 13, pp. 424-432.

[Vei66] Veinott, A. F., Jr., 1966. "The Status of Mathematical Inventory Theory," Management Sci., Vol. 12, pp. 745-777.

[Vin74] Vincke, P., 1974. "Problemes Multicriteres," Cahiers du Centre d' Etudes de Recherche Operationnelle, Vol. 16, pp. 425-439.

[Vit67] Viterbi, A. J., 1967. "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," IEEE Trans. on Info. Theory, Vol. IT-13, pp. 260-269.

[Vla08] Vladimirsky, A., 2008. "Label-Setting Methods for Multimode Stochastic Shortest Path Problems on Graphs," Math. of Operations Research, Vol. 33, pp. 821-838.

[WCG03] Wu, G., Chong, E. K. P., and Givan, R. L., 2003. "Congestion Control Using Policy Rollout," Proc. 2nd IEEE CDC, Maui, Hawaii, pp. 4825-4830.

[Wal47] Wald, A., 1947. Sequential Analysis, Wiley, N. Y.

[WeB99] Weaver, L., and Baxter, J., 1999. "Reinforcement Learning From State and Temporal Differences," Tech. Report, Department of Computer Science, Australian National University.

[WeP80] Weiss, G., and Pinedo, M., 1980. "Scheduling Tasks with Exponential Service Times on Nonidentical Processors to Minimize Various Cost Functions," J. Appl. Prob., Vol. 17, pp. l87-202.

[Wen14] Wen, Z., 1014. Efficient Reinforcement Learning with Value Function Generalization, Ph.D. Dissertation, Stanford University.

[WhH80] White, C. C., and Harrington, D. P., 1980. "Application of Jensen's Inequality to Adaptive Suboptimal Design," J. Optimization Theory Appl., Vol. 32, pp. 89-99.

[WhS89] White, C. C., and Scherer, W. T., 1989. "Solution Procedures for Partially Observed Markov Decision Processes," Operations Res., Vol. 30, pp. 791-797.

[Whi69] White, D. J., 1969. Dynamic Programming, Holden-Day, San Francisco, CA.

[Whi78] Whitt, W., 1978. "Approximations of Dynamic Programs I," Math. Operations Res., Vol. 3, pp. 231-243.

[Whi79] Whitt, W., 1979. "Approximations of Dynamic Programs II," Math. Operations Res., Vol. 4, pp. 179-185.

[Whi82] Whittle, P., 1982. Optimization Over Time, Wiley, N. Y., Vol. 1, 1982, Vol. 2, 1983.

[Whi88] Whittle, P., 1988. "Restless Bandits: Activity Allocation in a Changing World," J. of Applied Probability, pp. 287-298.

[Whi90] Whittle, P., 1990. Risk-Sensitive Optimal Control, Wiley, N. Y.

[Wil71] Willems, J., 1971. "Least Squares Stationary Optimal Control and the Algebraic Riccati Equation," IEEE Trans. on Automatic Control, Vol. 16, pp. 621-634.

[Wit66a] Witsenhausen, H. S., 1966. "Minimax Control of Uncertain Systems," Ph.D. Dissertation, Massachusetts Institute of Technology, Cambridge, MA.

[Wit66b] Witsenhausen, H. S., 1966. "A Comparison of Closed-Loop and Open-Loop Optimum Systems," IEEE Trans. Automatic Control, Vol. AC-11, pp. 619-621.

[Wit68] Witsenhausen, H. S., 1968. "Sets of Possible States of Linear Systems Given Perturbed Observations," IEEE Trans. Automatic Control, Vol. AC-13, pp. 556-558.

[Wit69] Witsenhausen, H. S., 1969. "Inequalities for the Performance of Suboptimal Uncertain Systems," Automatica, Vol. 5, pp. 507-512.

[Wit70] Witsenhausen, H. S., 1970. "On Performance Bounds for Uncertain Systems," SIAM J. on Control, Vol. 8, pp. 55-89.

[Wit71] Witsenhausen, H. S., 1971. "Separation of Estimation and Control for Discrete-Time Systems," Proc. IEEE, Vol. 59, pp. 1557-1566.

[Wol98] Wolsey, L. A., 1998. Integer Programming, Wiley, N. Y.

[WuB99] Wu, C. C., and Bertsekas, D. P., 1999. "Distributed Power Control Algorithms for Wireless Networks," unpublished report, available from the author's www site.

[YDR04] Yan, X., Diaconis, P., Rusmevichientong, P., and Van Roy, B., 2004. "Solitaire: Man Versus Machine," Advances in Neural Information Processing Systems, Vol. 17, pp. 1553-1560.

[YuB04] Yu, H., and Bertsekas, D. P., 2004. "Discretized Approximations for POMDP with Average Cost," Proc. of 20th Conference on Uncertainty in Artificial Intelligence, Banff, Canada.

[YuB12] Yu, H., and Bertsekas, D. P., 2012. "Weighted Bellman Equations and their Applications in Dynamic Programming," Lab. for Information and Decision Systems Report LIDS-P-2876, MIT.

[YuB15] Yu, H., and Bertsekas, D. P., 2015. "A Mixed Value and Policy Iteration Method for Stochastic Control with Universally Measurable Policies," Math. of Operations Research, Vol. 40, pp. 926-968.