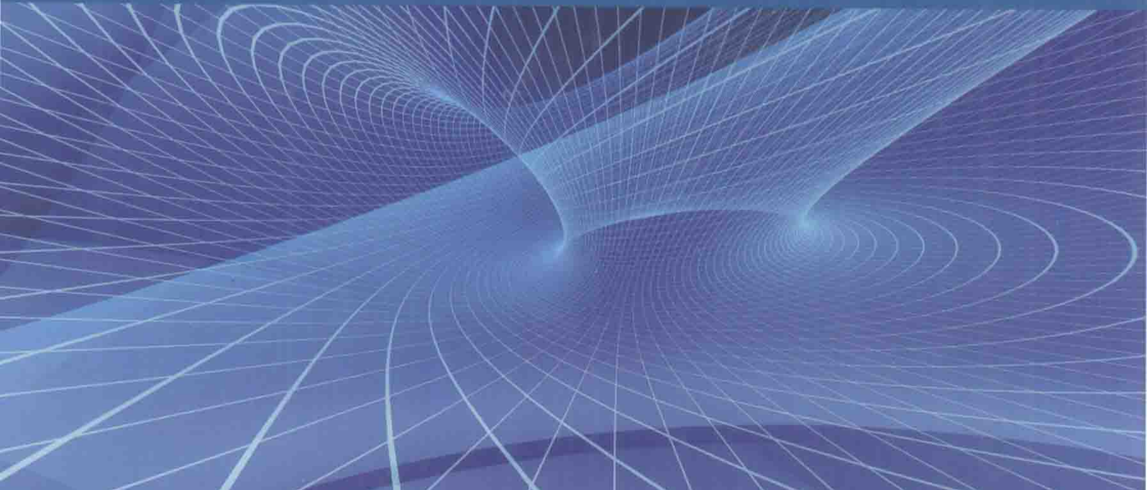
Concepts of Combinatorial Optimization

Edited by Vangelis Th. Paschos



ISTE

WILEY

Combinatorial Optimization
volume 1

Concepts of Combinatorial Optimization

常州大学图书馆
藏书章

Edited by
Vangelis Th. Paschos

ISTE

WILEY

First published 2010 in Great Britain and the United States by ISTE Ltd and John Wiley & Sons, Inc. Adapted and updated from *Optimisation combinatoire* volumes 1 to 5 published 2005-2007 in France by Hermes Science/Lavoisier © LAVOISIER 2005, 2006, 2007

Apart from any fair dealing for the purposes of research or private study, or criticism or review, as permitted under the Copyright, Designs and Patents Act 1988, this publication may only be reproduced, stored or transmitted, in any form or by any means, with the prior permission in writing of the publishers, or in the case of reprographic reproduction in accordance with the terms and licenses issued by the CLA. Enquiries concerning reproduction outside these terms should be sent to the publishers at the undermentioned address:

ISTE Ltd
27-37 St George's Road
London SW19 4EU
UK

www.iste.co.uk

John Wiley & Sons, Inc.
111 River Street
Hoboken, NJ 07030
USA

www.wiley.com

© ISTE Ltd 2010

The rights of Vangelis Th. Paschos to be identified as the author of this work have been asserted by him in accordance with the Copyright, Designs and Patents Act 1988.

Library of Congress Cataloging-in-Publication Data

Combinatorial optimization / edited by Vangelis Th. Paschos.
v. cm.
Includes bibliographical references and index.
Contents: v. 1. Concepts of combinatorial optimization
ISBN 978-1-84821-146-9 (set of 3 vols.) -- ISBN 978-1-84821-147-6 (v. 1)
1. Combinatorial optimization. 2. Programming (Mathematics)
I. Paschos, Vangelis Th.
QA402.5.C545123 2010
519.6'4--dc22

2010018423

British Library Cataloguing-in-Publication Data
A CIP record for this book is available from the British Library
ISBN 978-1-84821-146-9 (Set of 3 volumes)
ISBN 978-1-84821-147-6 (Volume 1)

Printed and bound in Great Britain by CPI Antony Rowe, Chippenham and Eastbourne.

MIX
Paper from responsible sources
FSC® C013604
FSC
www.fsc.org

Concepts of Combinatorial Optimization

Preface

What is combinatorial optimization? There are, in my opinion, as many definitions as there are researchers in this domain, each one as valid as the other. For me, it is above all the art of understanding a real, natural problem, and being able to transform it into a mathematical model. It is the art of studying this model in order to extract its structural properties and the characteristics of the solutions of the modeled problem. It is the art of exploiting these characteristics in order to determine algorithms that calculate the solutions but also to show the limits in economy and efficiency of these algorithms. Lastly, it is the art of enriching or abstracting existing models in order to increase their strength, portability, and ability to describe mathematically (and computationally) other problems, which may or may not be similar to the problems that inspired the initial models.

Seen in this light, we can easily understand why combinatorial optimization is at the heart of the junction of scientific disciplines as rich and different as theoretical computer science, pure and applied, discrete and continuous mathematics, mathematical economics, and quantitative management. It is inspired by these, and enriches them all.

This book, *Concepts of Combinatorial Optimization*, is the first volume in a set entitled *Combinatorial Optimization*. It tries, along with the other volumes in the set, to embody the idea of combinatorial optimization. The subjects of this volume cover themes that are considered to constitute the hard core of combinatorial optimization. The book is divided into three parts:

– Part I: Complexity of Combinatorial Optimization Problems;

– Part II: Classical Solution Methods;

– Part III: Elements from Mathematical Programming.

此为试读，需要完整PDF请访问：www.ertongbook.com

In the first part, Chapter 1 introduces the fundamentals of the theory of (deterministic) complexity and of algorithm analysis. In Chapter 2, the context changes and we consider algorithms that make decisions by "tossing a coin". At each stage of the resolution of a problem, several alternatives have to be considered, each one occurring with a certain probability. This is the context of probabilistic (or randomized) algorithms, which is described in this chapter.

In the second part some methods are introduced that make up the great classics of combinatorial optimization: branch-and-bound and dynamic programming. The former is perhaps the most well known and the most popular when we try to find an optimal solution to a difficult combinatorial optimization problem. Chapter 3 gives a thorough overview of this method as well as of some of the most well-known tree search methods based upon branch-and-bound. What can we say about dynamic programming, presented in Chapter 4? It has considerable reach and scope, and very many optimization problems have optimal solution algorithms that use it as their central method.

The third part is centered around mathematical programming, considered to be the heart of combinatorial optimization and operational research. In Chapter 5, a large number of linear models and an equally large number of combinatorial optimization problems are set out and discussed. In Chapter 6, the main simplex algorithms for linear programming, such as the primal simplex algorithm, the dual simplex algorithm, and the primal–dual simplex algorithm are introduced. Chapter 7 introduces some classical linear programming methods, while Chapter 8 introduces quadratic integer optimization methods. Chapter 9 describes a series of resolution methods currently widely in use, namely column generation. Chapter 10 focuses on polyhedral methods, almost 60 years old but still relevant to combinatorial optimization research. Lastly, Chapter 11 introduces a more contemporary, but extremely interesting, subject, namely constraint programming.

This book is intended for novice researchers, or even Master's students, as much as for senior researchers. Master's students will probably need a little basic knowledge of graph theory and mathematical (especially linear) programming to be able to read the book comfortably, even though the authors have been careful to give definitions of all the concepts they use in their chapters. In any case, to improve their knowledge of graph theory, readers are invited to consult a great, flagship book from one of our gurus, Claude Berge: *Graphs and Hypergraphs*, North Holland, 1973. For linear programming, there is a multitude of good books that the reader could consult, for example V. Chvátal, *Linear Programming*, W.H. Freeman, 1983, or M. Minoux, *Programmation mathématique: théorie et algorithmes*, Dunod, 1983.

Editing this book has been an exciting adventure, and all my thanks go, firstly, to the authors who, despite their many responsibilities and commitments (which is the lot of any university academic), have agreed to participate in the book by writing chapters in their areas of expertise and, at the same time, to take part in a very tricky exercise: writing chapters that are both educational and high-level science at the same time.

This work could never have come into being without the original proposal of Jean-Charles Pomerol, Vice President of the scientific committee at Hermes, and Sami Ménascé and Raphaël Ménascé, the heads of publications at ISTE. I give my warmest thanks to them for their insistence and encouragement. It is a pleasure to work with them as well as with Rupert Heywood, who has ingeniously translated the material in this book from the original French.

Vangelis Th. PASCHOS
June 2010

Table of Contents

Preface xiii
Vangelis Th. PASCHOS

PART I. COMPLEXITY OF COMBINATORIAL OPTIMIZATION PROBLEMS . . . 1

Chapter 1. Basic Concepts in Algorithms and Complexity Theory 3
Vangelis Th. PASCHOS

1.1. Algorithmic complexity 3
1.2. Problem complexity 4
1.3. The classes P, NP and NPO 7
1.4. Karp and Turing reductions 9
1.5. NP-completeness 10
1.6. Two examples of NP-complete problems 13
1.6.1. MIN VERTEX COVER 14
1.6.2. MAX STABLE 15
1.7. A few words on strong and weak NP-completeness 16
1.8. A few other well-known complexity classes 17
1.9. Bibliography 18

Chapter 2. Randomized Complexity 21
Jérémy BARBAY

2.1. Deterministic and probabilistic algorithms 22
2.1.1. Complexity of a Las Vegas algorithm 24
2.1.2. Probabilistic complexity of a problem 26
2.2. Lower bound technique 28
2.2.1. Definitions and notations 28
2.2.2. Minimax theorem 30
2.2.3. The Loomis lemma and the Yao principle 33

2.3. Elementary intersection problem 35
2.3.1. Upper bound. 35
2.3.2. Lower bound. 36
2.3.3. Probabilistic complexity. 37
2.4. Conclusion 37
2.5 Bibliography 37

PART II. CLASSICAL SOLUTION METHODS 39

Chapter 3. Branch-and-Bound Methods 41
Irène CHARON and Olivier HUDRY

3.1. Introduction. 41
3.2. Branch-and-bound method principles 43
3.2.1. Principle of separation 44
3.2.2. Pruning principles 45
3.2.3. Developing the tree. 51
3.3. A detailed example: the binary knapsack problem 54
3.3.1. Calculating the initial bound 55
3.3.2. First principle of separation. 57
3.3.3. Pruning without evaluation 58
3.3.4. Evaluation 60
3.3.5. Complete execution of the branch-and-bound method for finding only one optimal solution 61
3.3.6. First variant: finding all the optimal solutions 63
3.3.7. Second variant: best first search strategy 64
3.3.8. Third variant: second principle of separation 65
3.4. Conclusion 67
3.5. Bibliography 68

Chapter 4. Dynamic Programming 71
Bruno ESCOFFIER and Olivier SPANJAARD

4.1. Introduction. 71
4.2. A first example: crossing the bridge 72
4.3. Formalization. 75
4.3.1. State space, decision set, transition function 75
4.3.2. Feasible policies, comparison relationships and objectives 77
4.4. Some other examples 79
4.4.1. Stock management 79
4.4.2. Shortest path bottleneck in a graph 81
4.4.3. Knapsack problem 82
4.5. Solution 83
4.5.1. Forward procedure 84

4.5.2. Backward procedure 85
4.5.3. Principles of optimality and monotonicity 86
4.6. Solution of the examples 88
4.6.1. Stock management 88
4.6.2. Shortest path bottleneck 89
4.6.3. Knapsack 89
4.7. A few extensions 90
4.7.1. Partial order and multicriteria optimization 91
4.7.2. Dynamic programming with variables 94
4.7.3. Generalized dynamic programming 95
4.8. Conclusion 98
4.9. Bibliography 98

PART III. ELEMENTS FROM MATHEMATICAL PROGRAMMING 101

Chapter 5. Mixed Integer Linear Programming Models for Combinatorial Optimization Problems 103

Frédérico DELLA CROCE

5.1. Introduction 103
5.1.1. Preliminaries 103
5.1.2. The knapsack problem 105
5.1.3. The bin-packing problem 105
5.1.4. The set covering/set partitioning problem 106
5.1.5. The minimum cost flow problem 107
5.1.6. The maximum flow problem 108
5.1.7. The transportation problem 109
5.1.8. The assignment problem 110
5.1.9. The shortest path problem 111
5.2. General modeling techniques 111
5.2.1. Min-max, max-min, min-abs models 112
5.2.2. Handling logic conditions 113
5.3. More advanced MILP models 117
5.3.1. Location models 117
5.3.2. Graphs and network models 120
5.3.3. Machine scheduling models 127
5.4. Conclusions 132
5.5. Bibliography 133

Chapter 6. Simplex Algorithms for Linear Programming 135
Frédérico DELLA CROCE and Andrea GROSSO

6.1. Introduction 135
6.2. Primal and dual programs 135

6.2.1. Optimality conditions and strong duality 136
6.2.2. Symmetry of the duality relation 137
6.2.3. Weak duality 138
6.2.4. Economic interpretation of duality 139
6.3. The primal simplex method 140
6.3.1. Basic solutions 140
6.3.2. Canonical form and reduced costs 142
6.4. Bland's rule 145
6.4.1. Searching for a feasible solution 146
6.5. Simplex methods for the dual problem 147
6.5.1. The dual simplex method 147
6.5.2. The primal–dual simplex algorithm 149
6.6. Using reduced costs and pseudo-costs for integer programming 152
6.6.1. Using reduced costs for tightening variable bounds 152
6.6.2. Pseudo-costs for integer programming 153
6.7. Bibliography 155

Chapter 7. A Survey of some Linear Programming Methods 157
Pierre TOLLA

7.1. Introduction 157
7.2. Dantzig's simplex method 158
7.2.1. Standard linear programming and the main results 158
7.2.2. Principle of the simplex method 159
7.2.3. Putting the problem into canonical form 159
7.2.4. Stopping criterion, heuristics and pivoting 160
7.3. Duality 162
7.4. Khachiyan's algorithm 162
7.5. Interior methods 165
7.5.1. Karmarkar's projective algorithm 165
7.5.2. Primal–dual methods and corrective predictive methods 169
7.5.3. Mehrotra predictor–corrector method 181
7.6. Conclusion 186
7.7. Bibliography 187

Chapter 8. Quadratic Optimization in 0–1 Variables 189
Alain BILLIONNET

8.1. Introduction 189
8.2. Pseudo-Boolean functions and set functions 190
8.3. Formalization using pseudo-Boolean functions 191
8.4. Quadratic pseudo-Boolean functions (qpBf) 192
8.5. Integer optimum and continuous optimum of qpBfs 194
8.6. Derandomization 195

此为试读，需要完整PDF请访问：www.ertongbook.com

8.7. Posiforms and quadratic posiforms 196
8.7.1. Posiform maximization and stability in a graph 196
8.7.2. Implication graph associated with a quadratic posiform 197
8.8. Optimizing a qpBf: special cases and polynomial cases 198
8.8.1. Maximizing negative–positive functions 198
8.8.2. Maximizing functions associated with k-trees 199
8.8.3. Maximizing a quadratic posiform whose terms are associated with two consecutive arcs of a directed multigraph. 199
8.8.4. Quadratic pseudo-Boolean functions equal to the product of two linear functions 199
8.9. Reductions, relaxations, linearizations, bound calculation and persistence 200
8.9.1. Complementation 200
8.9.2. Linearization. 201
8.9.3. Lagrangian duality 202
8.9.4. Another linearization 203
8.9.5. Convex quadratic relaxation 203
8.9.6. Positive semi-definite relaxation 204
8.9.7. Persistence 206
8.10. Local optimum 206
8.11. Exact algorithms and heuristic methods for optimizing qpBfs 208
8.11.1. Different approaches 208
8.11.2. An algorithm based on Lagrangian decomposition 209
8.11.3. An algorithm based on convex quadratic programming 210
8.12. Approximation algorithms 211
8.12.1. A 2-approximation algorithm for maximizing a quadratic posiform 211
8.12.2. MAX-SAT approximation 213
8.13. Optimizing a quadratic pseudo-Boolean function with linear constraints 213
8.13.1. Examples of formulations 214
8.13.2. Some polynomial and pseudo-polynomial cases 217
8.13.3. Complementation 217
8.14. Linearization, convexification and Lagrangian relaxation for optimizing a qpBf with linear constraints 220
8.14.1. Linearization 221
8.14.2. Convexification 222
8.14.3. Lagrangian duality 223
8.15. ε-Approximation algorithms for optimizing a qpBf with linear constraints 223
8.16. Bibliography 224

Chapter 9. Column Generation in Integer Linear Programming 235
Irène LOISEAU, Alberto CESELLI, Nelson MACULAN and Matteo SALANI

9.1. Introduction. 235
9.2. A column generation method for a bounded variable linear programming problem 236
9.3. An inequality to eliminate the generation of a 0–1 column. 238
9.4. Formulations for an integer linear program 240
9.5. Solving an integer linear program using column generation 243
9.5.1. Auxiliary problem (pricing problem) 243
9.5.2. Branching 244
9.6. Applications 247
9.6.1. The p-medians problem 247
9.6.2. Vehicle routing 252
9.7. Bibliography 255

Chapter 10. Polyhedral Approaches. 261
Ali Ridha MAHJOUB

10.1. Introduction 261
10.2. Polyhedra, faces and facets. 265
10.2.1. Polyhedra, polytopes and dimension. 265
10.2.2. Faces and facets 268
10.3. Combinatorial optimization and linear programming 276
10.3.1. Associated polytope 276
10.3.2. Extreme points and extreme rays 279
10.4. Proof techniques 282
10.4.1. Facet proof techniques 283
10.4.2. Integrality techniques. 287
10.5. Integer polyhedra and min–max relations 293
10.5.1. Duality and combinatorial optimization 293
10.5.2. Totally unimodular matrices. 294
10.5.3. Totally dual integral systems 296
10.5.4. Blocking and antiblocking polyhedral 297
10.6. Cutting-plane method 301
10.6.1. The Chvátal–Gomory method. 302
10.6.2. Cutting-plane algorithm 304
10.6.3. Branch-and-cut algorithms. 305
10.6.4. Separation and optimization. 306
10.7. The maximum cut problem 308
10.7.1. Spin glass models and the maximum cut problem 309
10.7.2. The cut polytope. 310
10.8. The survivable network design problem 313
10.8.1. Formulation and associated polyhedron 314

10.8.2. Valid inequalities and separation 315
10.8.3. A branch-and-cut algorithm 318
10.9. Conclusion 319
10.10. Bibliography. 320

Chapter 11. Constraint Programming 325
Claude LE PAPE

11.1. Introduction 325
11.2. Problem definition 327
11.3. Decision operators 328
11.4. Propagation 330
11.5. Heuristics 333
11.5.1. Branching. 333
11.5.2. Exploration strategies. 335
11.6. Conclusion 336
11.7. Bibliography 336

List of Authors 339

Index 343

Summary of Other Volumes in the Series 347

PART I

Complexity of Combinatorial Optimization Problems

Chapter 1

Basic Concepts in Algorithms and Complexity Theory

1.1. Algorithmic complexity

In algorithmic theory, a problem is a general question to which we wish to find an answer. This question usually has parameters or variables the values of which have yet to be determined. A problem is posed by giving a list of these parameters as well as the properties to which the answer must conform. An instance of a problem is obtained by giving explicit values to each of the parameters of the instanced problem.

An algorithm is a sequence of elementary operations (variable affectation, tests, forks, etc.) that, when given an instance of a problem as input, gives the solution of this problem as output after execution of the final operation.

The two most important parameters for measuring the quality of an algorithm are: its *execution time* and the *memory space* that it uses. The first parameter is expressed in terms of the number of instructions necessary to run the algorithm. The use of the number of instructions as a unit of time is justified by the fact that the same program will use the same number of instructions on two different machines but the time taken will vary, depending on the respective speeds of the machines. We generally consider that an instruction equates to an elementary operation, for example an assignment, a test, an addition, a multiplication, a trace, etc. What we call the *complexity in time* or simply the *complexity* of an algorithm gives us an indication of the time it will take to solve a problem of a given size. In reality this is a function that associates an order of

Chapter written by Vangelis Th. PASCHOS.

此为试读,需要完整PDF请访问: www.ertongbook.com