

Springer Proceedings in Mathematics & Statistics

Antonio Sforza  
Claudio Sterle *Editors*

# Optimization and Decision Science: Methodologies and Applications

ODS, Sorrento, Italy, September 4–7,  
2017

 Springer

# **Springer Proceedings in Mathematics & Statistics**

Volume 217

## **Springer Proceedings in Mathematics & Statistics**

This book series features volumes composed of selected contributions from workshops and conferences in all areas of current research in mathematics and statistics, including operation research and optimization. In addition to an overall evaluation of the interest, scientific quality, and timeliness of each proposal at the hands of the publisher, individual contributions are all refereed to the high quality standards of leading journals in the field. Thus, this series provides the research community with well-edited, authoritative reports on developments in the most exciting areas of mathematical and statistical research today.

More information about this series at <http://www.springer.com/series/10533>

Antonio Sforza · Claudio Sterle  
Editors

# Optimization and Decision Science: Methodologies and Applications

ODS, Sorrento, Italy, September 4-7, 2017

 Springer

*Editors*

Antonio Sforza  
Department of Electrical Engineering and  
Information Technology  
University “Federico II” of Naples  
Naples  
Italy

Claudio Sterle  
Department of Electrical Engineering and  
Information Technology  
University “Federico II” of Naples  
Naples  
Italy

ISSN 2194-1009                      ISSN 2194-1017 (electronic)  
Springer Proceedings in Mathematics & Statistics  
ISBN 978-3-319-67307-3              ISBN 978-3-319-67308-0 (eBook)  
DOI 10.1007/978-3-319-67308-0

Library of Congress Control Number: 2017952505

Mathematics Subject Classification (2010): 90-06, 49-XX, 90Bxx, 49Kxx, 62Cxx, 91B06

© Springer International Publishing AG 2017

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Printed on acid-free paper

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

# Preface

Operations Research is known as the discipline founded on mathematical and quantitative methods aimed at determining optimal or near-optimal solutions to complex decision-making problems. The emphasis on the real applications highlights the great versatility and transversality of this discipline. Its solving approaches, methodologies and tools find application in many different fields and areas, notably in industrial and territorial systems. Hence, the interplay between researchers, practitioners and policy-makers plays a relevant role which is supported by conferences and workshops.

ODS2017, International Conference on Optimization and Decision Science, was the 47th annual meeting organized by the Italian Operations Research Society (AIRO) in Sorrento, Italy, September 4th–7th, 2017, in cooperation with the Department of Electrical Engineering and Information Technology (DIETI) of the University “Federico II” of Naples. The ODS2017 Programme and the Organizing Committee were composed of researchers from Italy, Europe and North America.

ODS2017 was addressed to the entire Operations Research and related scientific communities working in the wide field of optimization, problem-solving and decision-making methods. Its scope was presenting ideas and experiences on cutting-edge research topics, sharing knowledge, discussing challenging issues and results, and creating a point of contact to foster future collaborations among researchers and practitioners from various sectors (applied mathematics, computer science, engineering, economics), private and public companies, industries and policy-makers.

ODS2017 participants had the possibility either to submit a paper or an abstract on the conference research themes. All the contributions can be found in the conference e-book available at the website: [www.airoconference.it/ods2017](http://www.airoconference.it/ods2017). In this volume, the reader finds the collection of the invited lectures and research papers submitted and accepted for presentation at the conference after a peer-review process, made by experts in Operations Research and related fields.

The three invited lectures were:

- *Robust Network Control and Disjunctive Programming*, given by Prof. Daniel Bienstock, Department of Industrial Engineering and Operations Research, Columbia University, New York, USA.
- *From Mixed-Integer Linear to Mixed-Integer Bilevel Programming*, given by Prof. Matteo Fischetti, Department of Computer Science, University of Padova, Italy.
- *Data Science meets Optimization*, sponsored by the Association of the European Operations Research Societies, given by Prof. Patrick De Causmaecker, Department of Computer Science, University of Leuven, Belgium.

The submitted research papers spanned on the methodological and applicative themes proposed in the call for papers: Continuous and Global Optimization; Linear and Nonlinear Programming; Discrete and Combinatorial Optimization; Stochastic and Robust Optimization; Cutting, Packing and Scheduling, Multicriteria and Decision-Making, Energy optimization, Health Care, Data Science, Game Theory; Graph Theory and Network Optimization, Location, Routing; Urban Traffic, Freight Transportation; Logistics, Supply Chain Management; Railway and Maritime Systems Optimization; Telecommunication Networks, Critical Infrastructure Protection, Emergency Logistics, Emerging Applications.

The 60 accepted research papers are here organized in more aggregate sections, ranked in alphabetical order: Data Science, Health Care, Heuristics and Metaheuristics, Innovative Applications, Location, Multi-objective Optimization, Optimization Under Uncertainty, Packing and Cutting, Railway and Maritime Optimization, Routing, Scheduling. In each section, the papers are presented alphabetically by the last name of the first author. The classification has been done considering the main feature characterizing the paper, even if in several cases a paper could be framed in another section.

These articles highlight the impact that Operations Research methodologies and tools have in a society with increasing complexity-challenging problems and the cross-fertilization of ideas between theoretical and applicative fields. They exhibit the latest methods and techniques needed in solving a number of existing research problems while providing new open questions for further research investigations. It is expected that this research volume will be a valuable resource for experienced and young researchers.

As editors of this volume, we thank the invited lecturers and authors. Moreover, we express our sincere gratitude to the 71 researchers from around the world, who spent their valuable time for the review process, so contributing to improve the quality of the presented papers. We also express our thanks to Springer for support and cooperation in publishing the volume, bringing it to a nice form.

Finally, as conference chairs of ODS2017, we are thankful to the work team and students of the Department of Electrical Engineering and Information Technology, who actively helped in making the conference a success. We are also thankful to all

the institutions, agencies and enterprises that have supported and sponsored the event:

- University “Federico II” of Naples
- Polytechnic and Base Sciences School of Naples
- University of Sannio (Department of Engineering)
- University of Salerno  
(Department of Mathematics and Department of Industrial Engineering)
- IASI/CNR—Rome
- EURO (Association of the European OR Societies)
- IDIS Foundation—Science Center of Naples
- OPTIT S.r.l.
- ACTOR S.r.l.
- Ansaldo STS S.p.a.
- Lottomatica S.p.a.
- TecnoSistem S.p.a.
- Tekla S.r.l.
- ODS2017 Chairs and Volume Editors  
Antonio Sforza and Claudio Sterle

Naples, Italy

Antonio Sforza  
Claudio Sterle



# **Organization**

## **Conference Chairs**

Antonio Sforza, Department of Electrical Engineering and Information Technology,  
University “Federico II” of Naples

Claudio Sterle, Department of Electrical Engineering and Information Technology,  
University “Federico II” of Naples

# Contents

## Part I Invited Lectures

<b>Robust Network Control and Disjunctive Programming</b> . . . . .	3
Daniel Bienstock	
<b>Data Science Meets Optimization</b> . . . . .	13
Patrick De Causmaecker	
<b>From Mixed-Integer Linear to Mixed-Integer Bilevel Linear Programming</b> . . . . .	21
Matteo Fischetti	

## Part II Data Science

<b>Outperforming Image Segmentation by Exploiting Approximate K-Means Algorithms</b> . . . . .	31
Flora Amato, Mario Barbareschi, Giovanni Cozzolino, Antonino Mazzeo, Nicola Mazzocca and Antonio Tammaro	
<b>Approximate Decision Tree-Based Multiple Classifier Systems</b> . . . . .	39
Mario Barbareschi, Cristina Papa and Carlo Sansone	
<b>Look-Up Tables for Efficient Non-Linear Parameters Estimation</b> . . . . .	49
Stefano Marrone, Gabriele Piantadosi, Mario Sansone and Carlo Sansone	
<b>On the UTA Methods for Solving the Model Selection Problem</b> . . . . .	59
Valentina Minnetti	
<b>The Importance to Manage Data Protection in the Right Way: Problems and Solutions</b> . . . . .	69
Hassan Mokalled, Daniele Debertol, Ermete Meda and Concetta Pragliola	

**The Use of Configurational Analysis in the Evaluation of Real Estate Dynamics** . . . . . 83  
 Enrico G. Caldarola, Valerio Di Pinto and Antonio M. Rinaldi

**A Partitioning Based Heuristic for a Variant of the Simple Pattern Minimality Problem** . . . . . 93  
 Maurizio Boccia, Adriano Masone, Antonio Sforza and Claudio Sterle

**Part III Health Care**

**Patient–Centred Objectives as an Alternative to Maximum Utilisation: Comparing Surgical Case Solutions** . . . . . 105  
 Roberto Aringhieri and Davide Duma

**A Hierarchical Multi-objective Optimisation Model for Bed Levelling and Patient Priority Maximisation** . . . . . 113  
 Roberto Aringhieri, Paolo Landa and Simona Mancini

**Multi-Classifer Approaches for Supporting Clinical Diagnosis** . . . . . 121  
 Maria Carmela Groccia, Rosita Guido and Domenico Conforti

**Offline Patient Admission Scheduling Problems** . . . . . 129  
 Rosita Guido, Vittorio Solina and Domenico Conforti

**Stochastic Dynamic Programming in Hospital Resource Optimization** . . . . . 139  
 Marco Papi, Luca Pontecorvi, Roberto Setola and Fabrizio Clemente

**Part IV Heuristics and Metaheuristics**

**Column Generation Embedding Carousel Greedy for the Maximum Network Lifetime Problem with Interference Constraints** . . . . . 151  
 Francesco Carrabs, Carmine Cerrone, Ciriaco D’Ambrosio and Andrea Raiconi

**A Heuristic for Multi-attribute Vehicle Routing Problems in Express Freight Transportation** . . . . . 161  
 Luigi De Giovanni, Nicola Gastaldon, Ivano Lauriola and Filippo Sottovia

**Global Optimization Procedure to Estimate a Starting Velocity Model for Local Full Waveform Inversion** . . . . . 171  
 Bruno Galuzzi, Elena Zampieri and Eusebio Stucchi

**Ant Colony Optimization Algorithm for Pickup and Delivery Problem with Time Windows** . . . . . 181  
 M. Noubissi Tchoupo, A. Yalaoui, L. Amodeo, F. Yalaoui and F. Lutz

**Part V Innovative Applications**

**Initialization of Optimization Methods in Parameter Tuning for Computer Vision Algorithms** . . . . . 195  
 Andrea Bessi, Daniele Vigo, Vincenzo Boffa and Fabio Regoli

**Using OR + AI to Predict the Optimal Production of Offshore Wind Parks: A Preliminary Study** . . . . . 203  
 Martina Fischetti and Marco Fraccaro

**Mathematical Programming Bounds for Kissing Numbers** . . . . . 213  
 Leo Liberti

**Learning Greedy Strategies at Secondary Schools: An Active Approach** . . . . . 223  
 Violetta Lonati, Dario Malchiodi, Mattia Monga and Anna Morpurgo

**Comparison of IP and CNF Models for Control of Automated Valet Parking Systems** . . . . . 233  
 Abdullah Makkeh and Dirk Oliver Theis

**Part VI Location**

**A Districting Model to Support the Redesign Process of Italian Provinces** . . . . . 245  
 Giuseppe Bruno, Antonio Diglio, Alessia Melisi and Carmela Piccolo

**A Stochastic Maximal Covering Formulation for a Bike Sharing System** . . . . . 257  
 Claudio Ciancio, Giuseppina Ambrogio and Demetrio Laganá

**A Model and Algorithm for Solving the Landfill Siting Problem in Large Areas** . . . . . 267  
 Mariano Gallo

**Optimal Content Distribution and Multi-resource Allocation in Software Defined Virtual CDNs** . . . . . 275  
 Jaime Llorca, Antonia M. Tulino, Antonio Sforza and Claudio Sterle

**Facility Location with Item Storage and Delivery** . . . . . 287  
 Stefano Coniglio, Jörg Fliege and Ruth Walton

**A Shared Memory Parallel Heuristic Algorithm for the Large-Scale p-Median Problem** . . . . . 295  
 Igor Vasilyev and Anton Ushakov

## Part VII Multi-objective Optimization

<b>Robust Plasma Vertical Stabilization in Tokamak Devices via Multi-objective Optimization</b> . . . . .	305
Gianmaria De Tommasi, Adriano Mele and Alfredo Pironti	
<b>Performance Analysis of Single and Multi-objective Approaches for the Critical Node Detection Problem</b> . . . . .	315
Luca Faramondi, Gabriele Oliva, Roberto Setola, Federica Pascucci, Annunziata Esposito Amideo and Maria Paola Scaparra	
<b>Stable Matching with Multi-objectives: A Goal Programming Approach</b> . . . . .	325
Mangesh Gharote, Nitin Phuke, Rahul Patil and Sachin Lodha	
<b>On Relation of Possibly Efficiency and Robust Counterparts in Interval Multiobjective Linear Programming</b> . . . . .	335
Milan Hladík	
<b>Sustainable Manufacturing: An Application in the Food Industry</b> . . . . .	345
Maria Elena Nenni and Rosario Micillo	

## Part VIII Optimization Under Uncertainty

<b>The Optimal Energy Procurement Problem: A Stochastic Programming Approach</b> . . . . .	357
P. Beraldi, A. Violi, G. Carrozzino and M. E. Bruni	
<b>Best and Worst Values of the Optimal Cost of the Interval Transportation Problem</b> . . . . .	367
R. Cerulli, C. D'Ambrosio and M. Gentili	
<b>A Queueing Networks-Based Model for Supply Systems</b> . . . . .	375
Massimo De Falco, Nicola Mastrandrea and Luigi Rarità	
<b>Capital Asset Pricing Model—A Structured Robust Approach</b> . . . . .	385
Raquel J. Fonseca	
<b>On the Properties of Interval Linear Programs with a Fixed Coefficient Matrix</b> . . . . .	393
Elif Garajová, Milan Hladík and Miroslav Rada	
<b>Bounding Multistage Stochastic Programs: A Scenario Tree Based Approach</b> . . . . .	403
Francesca Maggioni and Elisabetta Allevi	
<b>A Polyhedral Study of the Robust Capacitated Edge Activation Problem</b> . . . . .	413
Sara Mattia	

**Performance Evaluation of a Push Merge System with Multiple Suppliers, an Intermediate Buffer and a Distribution Center with Parallel Channels: The Erlang Case** . . . . . 421  
 Despoina Ntio, Michael Vidalis, Stelios Koukoumialos and Alexandros Diamantidis

**A Push Shipping-Dispatching Approach for High-Value Items: From Modeling to Managerial Insights** . . . . . 431  
 Jean Respen and Nicolas Zufferey

**Part IX Packing and Cutting**

**Optimization Models for Cut Sequencing** . . . . . 443  
 Claudio Arbib, Pasquale Avella, Maurizio Boccia, Fabrizio Marinelli and Sara Mattia

**Bin Packing Problems with Variable Pattern Processing Times: A Proof-of-concept** . . . . . 453  
 Fabrizio Marinelli and Andrea Pizzuti

**Upper Bounds Categorization for Constrained Two-Dimensional Guillotine Cutting** . . . . . 461  
 Mauro Russo, Antonio Sforza and Claudio Sterle

**Part X Railway and Maritime Optimization**

**Some Complexity Results for the Minimum Blocking Items Problem** . . . . . 475  
 Tiziano Bacci, Sara Mattia and Paolo Ventura

**A MILP Algorithm for the Minimization of Train Delay and Energy Consumption** . . . . . 485  
 Teresa Montrone, Paola Pellegrini and Paolo Nobili

**A MILP Reformulation for Train Routing and Scheduling in Case of Perturbation** . . . . . 495  
 Paola Pellegrini, Grégory Marlière, Raffaele Pesenti and Joaquin Rodriguez

**Part XI Routing**

**The Impact of a Clustering Approach on Solving the Multi-depot IRP** . . . . . 507  
 Luca Bertazzi, Annarita De Maio and Demetrio Laganà

**Optimal Paths for Dual Propulsion Vehicles on Real Street Network Graphs** . . . . . 517  
 Giovanni Capobianco, Carmine Cerrone, Raffaele Cerulli and Giovanni Felici

**On the Forward Shortest Path Tour Problem** . . . . . 529  
 Francesco Carrabs, Raffaele Cerulli, Paola Festa and Federica Laureana

**A Flow Formulation for the Close-Enough Arc Routing Problem** . . . . . 539  
 Carmine Cerrone, Raffaele Cerulli, Bruce Golden and Rosa Pentangelo

**A Mesoscopic Approach to Model Route Choice in Emergency Conditions** . . . . . 547  
 Massimo Di Gangi and Antonio Polimeni

**Last-Mile Deliveries by Using Drones and Classical Vehicles** . . . . . 557  
 Luigi Di Puglia Pugliese and Francesca Guerriero

**A Scenario Planning Approach for Shelter Location and Evacuation Routing** . . . . . 567  
 Annunziata Esposito Amideo and Maria Paola Scaparra

**The Vehicle Routing Problem with Occasional Drivers and Time Windows** . . . . . 577  
 Giusy Macrina, Luigi Di Puglia Pugliese, Francesca Guerriero and Demetrio Laganà

**Part XII Scheduling**

**Preemptive Scheduling of a Single Machine with Finite States to Minimize Energy Costs** . . . . . 591  
 Mohammad Mohsen Aghelinejad, Yassine Ouazene and Alice Yalaoui

**An Optimization Model for the Outbound Truck Scheduling Problem at Cross-Docking Platforms** . . . . . 601  
 Antonio Diglio, Andrea Genovese and Carmela Piccolo

**Min-Max Regret Scheduling to Minimize the Total Weight of Late Jobs with Interval Uncertainty** . . . . . 611  
 Maciej Drwal

**Practical Solution to Parallel Machine Scheduling Problems** . . . . . 621  
 E. Parra

**Decomposition and Feasibility Restoration for Cascaded Reservoir Management** . . . . . 629  
 Wim van Ackooij, Claudia D’Ambrosio and Raouia Taktak

**Author Index** . . . . . 639

**Part I**  
**Invited Lectures**



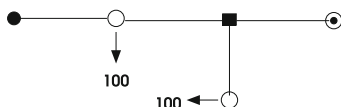
# Robust Network Control and Disjunctive Programming

Daniel Bienstock

**Abstract** The problems in this paper are motivated by studies concerning the deployment of storage (batteries) in power grids in order to mitigate stochastic behavior of renewables. From a mathematical standpoint, these problems can be viewed as two-stage adjustable robust optimization problems. We present a generic network-based robust optimization problem and describe a cutting-plane algorithm based on disjunctive programming for a specific application to power grid control.

## 1 Motivation

The problems we consider here can be motivated by a simple example.

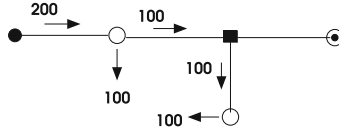


In this example we have a tree with nodes of four types: demand nodes (with known numerical demand shown in the figure), generator nodes (sources, depicted as solid circles) as well as two other types that will be described shortly. The edges of the tree have known capacities which are upper bounds on the *absolute* value of their flow. Finally for each generator node there is a function that describes the cost of generating any given amount at that node. A simple problem is that of satisfying demands at minimum cost without exceeding edge capacities. Note that total generation will equal total demand.

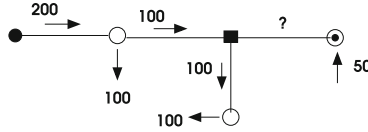
If the cost functions are convex we are dealing with a convex optimization problem. Suppose now that the problem plays out in *two* stages. First, we choose generation amounts so as to feasibly satisfy demands, as shown in the following figure:

---

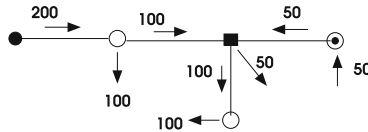
D. Bienstock (✉)  
Columbia University, New York City, NY, USA  
e-mail: dano@columbia.edu



In the second stage, however, the “uncontrollable” nodes become sources or sinks for flow:



We now have an infeasible (unbalanced) network with too much flow injection. Additionally the right-most edge of the tree may even carry too much flow. Putting aside this issue, we can address the imbalance between supply and demand by making use of the hitherto unmentioned fourth kind of node (the square node in the figure). These nodes, which model storage in power systems, will be used to absorb excess flow or to make up a shortage of flow.



The overall two-stage problem is that of choosing generation amounts, at minimum cost, so that for *any* flow injection at the nodes, from within a modeled set of such injections, there is a corresponding reaction at the square nodes (also chosen within limitations) so that the resulting flow vector does not exceed any edge capacities.

## 1.1 Formulation

We now provide a mathematical formulation for the type of problem described above. For simplicity we focus on a single-period model. We are given an undirected network  $\mathcal{G}$  with node-set  $\mathcal{N}$ , where each edge  $e$  has a capacity  $F_e^{\max} > 0$ . In addition, at every node  $k$  we have the following data and decision variables:

- $P_k^g$  = amount of generation at  $k$ , a variable, constrained by  $0 \leq P_k^g \leq P_k^{g,\max}$  where  $P_k^{g,\max}$  is a given quantity, possibly equal to zero (indicating a non-generating node). The *cost* of generating  $x$  units at  $i$  is given by a function  $\kappa_k(x)$ .
- $P_k^d$  = demand at  $i$  (data).
- $w_k$  = uncontrollable injection at  $k$ . This is an uncertain quantity that gives rise to the robust optimization problem. We assume  $w \in \mathcal{W}$ , where  $\mathcal{W} \subseteq \mathbb{R}^{\mathcal{N}}$  is a given set.
- $u_k$  = control injection at  $k$ . We constrain  $u \in \mathcal{U}$ , where  $\mathcal{U} \subseteq \mathbb{R}^{\mathcal{N}}$  is a given set.

The problem in question can now be written as:

$$\min_{P^g} \sum_k \kappa_k(P_k^g) \quad (1)$$

$$\text{s.t. } 0 \leq P_k^g \leq P_k^{g,\max} \quad \text{for all nodes } k, \quad (2)$$

and for each  $w \in \mathcal{W}$ , there exists  $u \in \mathcal{U}$  so that

the following holds:

$$\sum_k (P_k^g + w_k - u_k - P_k^d) = 0 \quad (\text{total supply} = \text{total demand}) \quad (3)$$

$$|\text{flow on edge } e| \leq F_e^{\max}, \quad \text{for all edges } e. \quad (4)$$

Thus, the model computes a minimum-cost generation plan that feasibly meets demands under all uncertain injections  $w$ , using the controls  $u$  as a second-stage correction.

Constraint (4) merits a discussion. Given an edge  $e$ , its flow under a given vector  $w$  and control  $u$  will be a function of  $P^g$ ,  $P^d$ ,  $w$  and  $u$ . In traditional network flow models this function would be a feature that we can control as part of the optimization process, that is to say, flow can be “routed”. However,

- In the linearized model for electrical power flows, the flow on  $e$  is a linear function of the form  $\pi_e^T(P^g - P^d + w - u)$  for a certain known vector  $\pi_e \in \mathbb{R}^{\mathcal{N}}$ . Thus, flow *cannot* be routed.
- In the (nontrivial and important) case where the network  $\mathcal{G}$  is a tree, we can explicitly write the flow. Namely if  $e$  has endpoints  $i$  and  $j$ , say, and  $T_i$  is the tree in  $\mathcal{G} - e$  containing  $i$ , then the flow on  $e$  in the  $i$  to  $j$  direction equals  $\sum_{k \in T_i} (P_k^g - P_k^d + w_k - u_k)$ .

Problem (1–4) belongs to a much broader family of problems, with general linear constraints as opposed to network-based constraints such as (4), which have been termed *two-stage adjustable optimization* problems. Previous work has focused on specific *policies* for the second-stage decisions, which in the problem above are represented by the variables  $u$ . For example an affine policy would amount to setting  $u = \lambda_0 + \lambda^T w$  where  $\lambda_0 \in \mathcal{R}$  and  $\lambda \in \mathbb{R}^{\mathcal{N}}$  are decision variables. Two-stage adjustable optimization lies at the core of robust optimization theory and we lack the space for a proper literature review. However a very apt citation with a literature survey is [1]. A different perspective on two-stage adjustable problems is that they may also be viewed as *bilevel optimization* problems, which have acquired a well-earned reputation for extreme difficulty. A recent contribution that includes computation is [2]. Also see [3] which uses intersection cuts. Finally, a different robust network flow problem is considered in [4].

## 1.2 NP-Hardness of General Problem

Let us return to problem (1–4). The *feasibility* problem is the following: given a vector  $P^g \in \mathbb{R}^{\mathcal{N}}$  satisfying (2), is it feasible? That is to say, is it the case that for every  $w \in \mathcal{W}$ , there exists  $u \in \mathcal{U}$  so that (3–4) holds?

It is known that in the general two-stage adjustable optimization context the corresponding feasibility problem is in fact NP-complete. See [5, 6]. However, our setting is quite specific. Nevertheless, we have:

**Theorem 1** *The feasibility problem for (1–4) is strongly NP-complete even if the underlying network  $\mathcal{G}$  is a star,  $\mathcal{W}$  is a polyhedron and  $\mathcal{U}$  is a hypercube.*

## 2 Batteries and Renewables in Power Grids

We now describe the concrete problem we are studying (see [7] for additional material, [8] for related work, and [9] for broader background). The problem differs from the above model in several ways. First, it covers  $T$  time periods of equal length (of unit length, to simplify notation) with all decisions made at time zero on the basis of forecasts.

- At time zero we compute the generation at node  $k$  in period  $t$ ,  $P_{k,t}^g$ , as well as parameters  $\lambda_{ij}^t \geq 0$  for a linear control given below.
- At time  $t$ , the (uncontrollable) injection at bus  $k$  is of the form  $w_{k,t}^f + w_{k,t}$  where  $w_{k,t}^f$  is a forecast. We rely on a *robust* model  $\mathcal{W}$  for the disposition of the vector of all the deviations  $w_{k,t}$ . The model is described below. We also assume *known* demands  $P_{k,t}^d$  for all nodes  $k$  and periods  $t$ .
- At time  $t$ , the output of control (battery) node  $i$  will be of the form  $-\sum_j \lambda_{ij}^t w_{j,t}$ . This implies a *linear* control as opposed to the general control envisioned in Sect. 1. The implication of this control scheme is that with  $\lambda_{ij}^t > 0$ , if e.g.  $w_{j,t} < 0$  then the battery responds to a decrease in renewable output by increasing its output. Linear control schemes are appealing in part because of the increased tractability of the resulting optimization problem, but also because they result in simpler and more intuitive control policies.
- Finally, we have an additional model that applies to each battery. It covers feasible actions for the battery under appropriate battery chemistry assumptions. In essence this model limits the set of possible injections  $-\sum_j \lambda_{ij}^t w_{j,t}$  for a battery at node  $i$ , over all  $1 \leq t \leq T$  and all  $w \in \mathcal{W}$ . Note that for a given time period  $t$ , the sum of all power injections at a node  $i$  holding a battery (through period  $t$ ) will be of the form  $-\sum_{h=1}^t \sum_j \lambda_{ij}^h w_{j,h}$ . This expression shows that the state of the battery at any time depends on actions on all prior periods.

We can now write an initial formulation, by  $\Lambda_t$  the vector of all  $\lambda_{k,t}$  (and  $\Lambda$  is the concatenation of all  $\Lambda_t$ ), and with similar interpretation for  $P_t^g$ ,  $P_t^d$ ,  $w_t^f$  and  $w_t$ .

This formulation omits the specification of battery constraints, which we will provide later.

$$\min_{P^g, \Lambda \geq 0} \sum_k \kappa_{k,t}(P_{k,t}^g) \quad (5)$$

$$\text{s.t. } 0 \leq P_{k,t}^g \leq P_{k,t}^{g,\max} \quad \text{for all nodes } k, \text{ and all periods } t \quad (6)$$

and such that for each  $w \in \mathcal{W}$

the following holds:

$$\sum_k (P_{k,t}^g + w_{k,t}^f + w_{k,t} - \sum_j \lambda_{k,j}^t w_{j,t} - P_{k,t}^d) = 0 \quad (7)$$

$$|\pi_e^T (P_t^g + w_t^f + w_t - \sum_j \lambda_j^t w_t - P_t^d)| \leq F_{e,t}^{\max}, \quad \text{for all edges } e \text{ and periods } t. \quad (8)$$

Constraint (7) states that total supply = total demand during period  $t$ . Constraint (8) indicates that the capacity of every edge is never exceeded.

## 2.1 Uncertainty Modeling

A popular technique used to model uncertainty, in the power engineering literature, is to rely on *scenario-driven* models. While the resulting optimization problems are simple, they can also become quite large if a guarantee is sought, and as a result authors often heuristically rely on small scenario sets. Another class of models that have become popular are chance-constrained optimization problems using a normality assumption. While in single-period models this is a plausible assumption (see e.g. [8]) in a multi-period model one should expect correlation (both space- and time-wise).

We rely instead on a *robust* model. So-called ‘‘uncertainty budgets’’ models have received attention in robust optimization community. See e.g. [10]. In the case of the problem in this discussion one would rely on constraints of the form

$$|w_{k,t}| \leq \gamma_{t,k}, \quad \text{all } t \text{ and } k \quad (9)$$

$$\sum_k (\gamma_{k,t})^{-1} |w_{k,t}| \leq \Gamma^t \quad \text{all } t \quad (10)$$

to model the renewable deviations, where the  $\gamma$  and  $\Gamma$  are parameters estimated from data. Note that the resulting model is *symmetric* around zero.

However, we wish to allow for *asymmetry* in renewable deviations. As a generalization of (9, 10) we use a model given by nonnegative matrices  $K^+$  and  $K^-$ , and a vector  $b$ . The set  $\mathcal{W}$  of allowable deviations is given by the constraint

$$K^+ w^+ + K^- w^- \leq b \quad (11)$$

Here,  $w^+$  ( $w^-$ ) is the vector with entries  $w_{k,t}^+ = \max\{w_{k,t}, 0\}$  (resp.,  $w_{k,t}^-$ ). Notice that (11) is certainly not linear. In fact a fully linear description would be exponentially large. On the other hand, (11) is “convex to the origin,” that is to say if  $w$  satisfies (11) then for any vector  $\theta$  with entries in  $[0,1]$ , the vector  $\theta \bullet w$  also satisfies (11). Finally, note that (11) also allows for correlation across time and space, a desired feature.

At this point we can state a useful property.

**Lemma 1** *Let  $\mathcal{W}$  be given by (11) Suppose  $(\hat{P}^g, \hat{\Lambda})$  is a candidate solution for problem (5–8). Suppose that for some edge  $e$  there exists  $w \in \mathcal{W}$  such that (8) is violated. Then, without loss of generality, either  $w_{k,t} \geq 0$  for all  $k$  and  $t$  or  $w_{k,t} \geq 0$  for all  $k$  and  $t$ .*

The implication of this result is that, in terms of constraints (8), the separation problem reduces to a number of linear programs.

## 2.2 Battery Modeling

Battery chemistry can be complex and in particular performance of a battery can be state-dependent. Here, “state” means the chemical charge state of the battery (i.e. what percentage of maximum chemical charge it holds). And performance will mean two things: first, how much (instantaneous) power can be removed from the battery or input into the battery. And second, the charge/discharge efficiency, which is to say the change in internal (i.e. chemical) charge as a function of the amount of power added or removed to the battery, which is normally described as charge or discharge efficiency.

Previous models in the literature assume a single charging efficiency  $0 < \eta_c < 1$  and a single discharging efficiency  $0 < \eta_d < 1$ . If an instantaneous power injection of  $P$  units of power into the battery takes place ( $P < 0$  means extraction of power) then the internal charge changes, per unit time, by

$$\chi(P) \doteq \eta_c P^+ - \eta_d^{-1} P^-$$

where  $P^+$  and  $P^-$  are the positive and negative parts of  $P$ , respectively. Clearly the function  $\chi(P)$  is nonlinear, and in an optimization model where the quantity  $P$  is a decision variable (as is the case in our model) one needs an appropriate way to model the complementarity condition  $P^+ P^- = 0$  and some authors have relied on integer variables for this purpose.

Here we will also assume the single charge/discharge efficiency model (a more general model is considered in [7]). Consider a given battery at node  $i$ , say. We denote by  $E^0$  the initial charge, and by  $E^{\min}$  ( $E^{\max}$ ) be the minimum (resp., maximum) allowable charge (dropping the index  $i$  for simplicity). The constraints we impose are

(a) For any  $w \in \mathcal{W}$ , and any time period  $t$ ,

$$E^{\min} \leq E^0 + \sum_{h=1}^t \chi \left( \sum_j \hat{\lambda}_{ij}^t w_{j,t} \right) \leq E^{\max} \quad (12)$$

The rationale for this condition is that the expression being bounded is the energy state of the battery at time  $t$ , assuming deviations  $w$ .

(b) We assume a piecewise-constant model given by breakpoints  $E^{\min} = c_0 < c_1 < \dots < c_K = E^{\max}$ , such that in each interval  $(c_s, c_{s+1})$  the output of the battery is constrained. Specifically, if battery charge lies in an interval  $(c_s, c_{s+1})$  there is a maximum  $I_s^{\max}$  amount of power that can be injected into the battery (and, similarly, a maximum  $X_s^{\max}$  that can be extracted from the battery). We will term this condition the **battery speed** constraint.

### 2.2.1 Separation Problem for Battery Constraints

We can now state, in words, the feasibility constraints for batteries, which up to now have been missing from model (5–8). We will focus in particular on the battery speed constraints.

Consider a given battery (we omit node indices to simplify notation). Suppose that is the initial energy level. the energy level. Let  $t$  be a time period, and let  $(\hat{P}^g, \hat{\Lambda})$  be a candidate solution. Then for any  $w \in \mathcal{W}$ , and any  $0 \leq s < K$  the constraint reads: **If** at the start of period  $t$ , battery charge lies in  $(c_s, c_{s+1})$ , **then**

$$-X_s^{\max} \leq \sum_j \hat{\lambda}_{ij}^t w_{j,t} \leq I_s^{\max} \quad (13)$$

Note that we are unable to write in a compact form the first part of this statement. Nevertheless, we will show next that we are able to check for violation of the condition in an efficient manner.

To see that this is the case, assume that there is a  $w \in \mathcal{W}$  such that the first part of the condition holds (the “If” statement) and yet (13) does not hold. To fix ideas, suppose that  $E^{\text{init}} > c_s$ , i.e. battery charge has moved to a lower energy interval and that at time  $t$  it is the second constraint in (13) that is violated (all other cases are similar). This means that the battery under consideration is receiving *too much charge* given its energy state.

Suppose we replace the vector  $w$  with a new vector  $\bar{w}$  defined by  $\bar{w}_{k,h} = w_{k,h}^-$  for all  $k$  and  $h < t$ ,  $\bar{w}_{k,t} = w_{k,t}^+$  for all  $k$ , finally  $\bar{w}_{k,h} = 0$  for all  $k$  and  $h > t$ . Then  $\bar{w}$  satisfies (11) i.e.  $\bar{w} \in \mathcal{W}$ . Moreover, under deviations  $\bar{w}$  the battery will only discharge in periods prior to  $t$ , and so will certainly end in some interval  $(c_r, c_{r+1})$  with  $r \leq s$ .

Hence for some scale value  $0 \leq \alpha \leq 1$ , the vector  $w^*$  given by  $w_{k,t}^* = \alpha \bar{w}_{k,h}$  for all  $k$  and  $h < t$  and  $w_{k,h}^* = \bar{w}_{k,h}$  for all  $k$  and  $h \geq t$  will *also* be contained in  $\mathcal{W}$  and will satisfy that

- (1) battery charge at the start of period  $t$  lies in the range  $(c_s, c_{s+1})$ ,
- (2)  $w_{k,h}^* \leq 0$  for all  $k$  and  $h \leq t$ . In other words under deviations  $w^*$ , the first  $t - 1$  time periods have nonpositive forecast errors (and in particular monotone),
- (3) In period  $t$  we have  $w_{k,t}^* \geq 0$  for all  $k$  and so all batteries charge.
- (4) At the battery of interest,

$$\sum_j \hat{\lambda}_{i,j}^t w_{j,t}^* > I_s^{\max}. \quad (14)$$

In other words, if the battery speed condition is violated we can restrict the search for  $w$  by restricting our attention to vectors satisfying (1)–(4). Since we know the signs of all entries of such vectors, the nonlinear model (11) becomes *linear*, and checking for condition (14) reduces to solving a linear program.

Moreover, if an offending vector  $w^*$  as above is found, then, assuming that  $(\hat{P}^g, \hat{\Lambda})$  is an *extreme point* of the current formulation, we can find a cut that separates  $(\hat{P}^g, \hat{\Lambda})$  from the current feasible region. The reason that this is the case is that any feasible  $(P^g, \Lambda)$  must either start period  $t$  with energy level at most  $c_s$ , or at least  $c_{s+1}$  or satisfy

$$\sum_j \lambda_{i,j}^t w_{j,t}^* \leq I_s^{\max}.$$

This is a linear condition, and note that (see e.g. (12)) the energy state under deviations  $w^*$ , at time  $t$ , is given by

$$E^0 + \sum_{h=1}^{t-1} \left( \sum_j \lambda_{i,j}^h \eta_d^{-1} w_{j,h}^* \right) + \eta_c \left( \sum_j \hat{\lambda}_{i,j}^t w_{j,t}^* \right)$$

We thus obtain a three-way disjunction (of linear inequalities) all of whose terms are violated by  $(\hat{P}^g, \hat{\Lambda})$ ; consequently a disjunctive-cut [11] can be used to separate  $(\hat{P}^g, \hat{\Lambda})$ .

### 3 A Numerical Experiment

The algorithm was implemented using Gurobi [12] as the LP solver. For these tests we used the winter peak Polish grid from MATPOWER [13], with 2746 nodes, 3514 edges, 388 generators, base load (approx.) 24.8 GW, 32 wind farms, with forecast output 4.5 GW and 32 batteries, with total initial charge approx. 3.2 GWh. We used the uncertainty budgets robustness model with an implied forecast error of up to 8.9%. In the runs below, loads increase (in similar proportions) in the first six periods. In our implementation, the initial formulation includes all line constraints and flow balance equations for the nominal case (no forecast errors). In the following table, “n” and “m” indicated the number of variables and constraints in the master formulation



at termination (but including some preprocessing). The running time is primarily accrued by solving linear programs, whose size grows proportional to the number of periods. The number of iterations appears nearly constant.

T	n	m	Cost	Iterations	Time (s)
6	20940	57519	7263172	20	366
8	27920	76651	9738804	17	442
10	34900	95825	12260289	19	670
12	41880	114995	14784028	18	848

**Acknowledgements** This work has been funded in part by grants from LANL (Grid Modernization), DARPA (Radics) and DTRA.

## References

1. El Housni, O., Goyal, V.: Piecewise static policies for two-stage adjustable robust optimization. *Math. Prog. A* (2017) (To appear)
2. Tahernejad, S., Ralphs, T.K., DeNegre, S.T.: A branch-and-cut algorithm for mixed integer bilevel linear optimization problems and its implementation, optimization online (2017). [http://www.optimization-online.org/DB\\_HTML/2017/04/5977.html](http://www.optimization-online.org/DB_HTML/2017/04/5977.html). Accessed 06 May 2017
3. Fischetti M., Ljubić I., Monaci M., Sinnl, M.: Intersection cuts for bilevel optimization. In: *Proceedings of IPCO*, pp. 77–88 (2016)
4. Atamtürk, A., Zhang, M.: Two-stage robust network flow and design under demand uncertainty. *Oper. Res.* **55**, 662–673 (2007)
5. Feige, U., Jain, K., Mahdian, M., Mirrokni, V.: Robust combinatorial optimization with exponential scenarios. In: *Proceedings of IPCO*, pp. 439–453 (2007)
6. Minoux, M.: On robust maximum flow with polyhedral uncertainty sets. *Optim. Lett.* **3**, 367–376 (2009)
7. Bienstock, D., Matke, M., Muñoz, G., Yang, S.: Robust linear control of nonconvex battery operation in transmission systems. [arXiv:1610.09432](https://arxiv.org/abs/1610.09432) (2016)
8. Bienstock, D., Chertkov, M., Harnett, S.: Chance-constrained DC-OPF. *SIAM Rev.* **56**, 461–495 (2014)
9. Bienstock, D.: *Electrical Transmission System Cascades and Vulnerability: An Operations Research Viewpoint*. SIAM-MOS Series on Optimization (2015)
10. Bertsimas, D., Brown, D.B., Caramanis, C.: Theory and applications of robust optimization. *SIAM Rev.* **53**(2011), 464–501 (2011)
11. Balas, E.: Disjunctive programming: cutting planes from logical conditions. In: Mangasarian, O.L., et al. (eds.) *Nonlinear Programming 2*, pp. 279–312. Academic Press, London (1975)
12. GUROBI: Gurobi Optimizer. <http://www.gurobi.com/>
13. Zimmerman, R.D., Murillo-Sanchez, C.E., Thomas, R.J.: MATPOWER: steady-state operations. Planning, and analysis tools for power systems research and education. *IEEE Trans. Power Syst.* **26**, 12–19 (2011)

# Data Science Meets Optimization

Patrick De Causmaecker

## 1 Introduction

Data science and optimisation have evolved separately over several decades. In [16], Tony Hey and his co-authors, inspired by the late, Turing award winning, Jim Gray picture data science as a fourth paradigm in scientific evolution. After the empirical branch, the theoretical and the computational branch, data exploration comes to the scene. Historically, this evolution spreads over literary thousand years and most of research domains can be traced back a long time in history. An interesting overview on combinatorial optimisation is [23]. Interesting cases are e.g. the Steiner problem [1, 4] and Kepler's conjecture [15].

Here we will review some recent developments in the application of data science to support heuristics for combinatorial optimisation problems as well as the use of heuristics in a data science context.

## 2 Data Science for Optimisation

Optimisation is the discipline of finding values of variables that optimise some goal function(s) within a given domain. The description of the problem is typically not more than a couple of pages long. Its complexity status may or may not be known. In practical situations, the problem is not in the complexity status, but in the expected execution time for a specific instance.

Data science may enter in various ways. The difficulty of problems often strongly depends on the properties of the problem instance. Producing optimal solutions in

---

P. De Causmaecker (✉)  
CODES, Department of Computer Science, KU Leuven/KULAK,  
E. Sabbelaan 53, 8500 Kortrijk, Belgium  
e-mail: patrick.decausmaecker@kuleuven.be

a guaranteed amount of computation time is often not possible. The problem shifts from finding the optimal solution towards finding an as good as possible solution in an acceptable amount of time for most of the problem instances that will be thrown at the system.

## **2.1 DFO Example: Using Data Science in Algorithm Construction**

A first example with remarkable successes in recent years is automated algorithm construction. Automated algorithm construction is one possibility in a set of automated mechanisms allowing data to be used for optimisation. Available are a set of problem instances and a formal description of the problem. The aim is to produce an algorithm that solves the problem as efficiently as possible. One could argue that techniques such as algorithm selection and algorithm tuning are in fact algorithm construction techniques. In algorithm selection, a set of algorithms for the problem is defined and the aim of the automated construction phase is limited to learning a predictor from the data that selects the best algorithm to solve a given problem. In algorithm tuning, one algorithm is given with a number of discrete and continuous parameters. The aim of the automated phase is to find the best setting of the parameters for the expected distribution of the problem instances of which the data is supposed to be a representative sample.

An obvious argument against is that in selection, as well as in tuning, complete algorithms have been designed and only a limited number of decisions are left to the automated construction module. In the other extreme, no constraints except the available operations at the machine or programming language level should exist.

As an example we discuss the approach in [2] for multi objective evolutionary algorithm (MOEA) construction. The authors present a general conceptual view of an MOEA, and prior to the automated approach, demonstrate how a number of well known MOEA's from literature can be described. MOEA's are seen as combinations of lower-level components such as preference relations and archives. The resulting framework extends the number of algorithms that can be instantiated as well as the number of MOEA approaches that can be designed. Given this large design space, an efficient navigator is needed to find the best combination of components for a given problem. The authors use the parameter tuning package *irace*[20] for this purpose. *irace* accepts discrete and continuous parameters and can hence be configured as a selector of components provided a specific pattern is given. *irace* then only has to decide which component to insert where. The authors design new databases of instances for several problems and study the behaviour of the constructor. The same problem with different instances turns out to lead to significantly different algorithms. The constructed MOEA's outperform existing algorithms, even after these existing algorithms have been tuned by *irace*.

This example uses expert human knowledge formalised in an algorithm template and a black box analysis of a representative data set of instances. Some other examples are [3, 6, 18, 19, 21, 24]. The algorithm components in this approach are general and not problem specific. In the next section we discuss an approach that allows algorithm designers to insert knowledge about the problem. Optimisation is not only linked to data but also to expert knowledge.

## 2.2 *DFO Example: Using Data Science While Engineering an Algorithm*

Consider the problem of designing a multi-neighbourhood local search algorithm. Our running example will be a vehicle routing problem (VRP). A multi-neighbourhood local search algorithm is an algorithm that moves from one acceptable solution for the VRP to another according to a number of neighbourhoods. The local search could simply be hill climbing or it could employ a mechanism allowing it to escape from local optima. Selection of a neighbourhood may be (weighted) random selection or any intelligent, machine learning based, mechanism. A specific neighbourhood may reflect expert understanding of the problem domain or design experience. Designers will come up with a multitude of possible neighbourhoods. The question that remains is which neighbourhoods to actually use.

This is an algorithm construction problem as we saw in Sect. 2.1, with the weights as parameters. Another option is a hyper-heuristic approach [5] where the weights evolve during the search guided by machine learning. All these are black box approaches. The present example shows how to open the box.

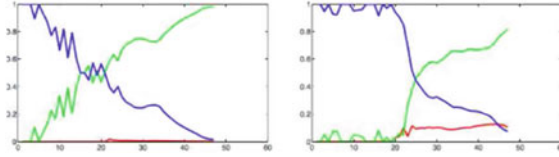
The target is to offer support to algorithm designers. They will experiment with neighbourhood combinations on a set of sample problems. The information retrieved will be fragmented and no *why-questions* will be answered.

To allow observation of neighbourhood behaviour during the search [9] uses code inserts to register neighbourhood behaviour. Observables are goal function values, times taken, . . . Evolution of these observables for each neighbourhood in the course of the search is stored. Inspection of the resulting file may a.o. reveal varying performance of neighbourhoods during the search.

The data plotted against time may not offer the full picture. Local search algorithms often wander around in extended valleys before discovering higher hills. A few observations in such a region may be sufficient to identify effective neighbourhoods. More interesting information comes when the goal function approaches its optimum.

The authors of [9] introduce the value of the goal function as the basis of a plot. This goal function value is taken from a, conveniently discretised, finite interval.

The collected data reflects the events during one run. The following features on each neighbourhood  $N$  include number of times used, number and amount of improvements, no effects, worsening. Each of these features is plotted against the



**Fig. 1** Visualization of logged numbers of improvement, no effect and worsening for two neighbourhoods. The x-axis represents solution quality (the larger the value, the better the corresponding solution is). The y-axis represents values of the three observables

goal function value taken from a VNS algorithm. After dividing the goal function range interval into regions, [9] collects the quantities in a vector characterising each neighbourhood. See Fig. 1 for an example.

The procedure described now serves two purposes. It allows visualisation of neighbourhood behaviour for algorithm developers to select a diverse set of limited size or to identify properties of missing neighbourhoods. It allows analysing an existing algorithm automatically. If one supposes that neighbourhoods with similar profiles behave similarly, these characteristics may allow clustering neighbourhoods and selecting one from each cluster. This may lead to more efficient tuning possibilities.

In [9], such a study was performed. A set of 42 neighbourhoods was reduced to 9 and it was shown that the result of applying *iraces* significantly outperformed the 42-neighbourhood version.

### 3 Optimisation for Data Science

#### 3.1 OFD Example: Community Detection in Graphs

Large graphs have many applications in network analysis. Examples are records of cell phone calls, e-mail connections, influencer graphs in social media, .... The scales of these networks vary from rather small (up to 1000 nodes) over medium (say 50000 nodes) up to very large graphs ( $> 10^6$  nodes). For a review see [13, 14].

One subject of interest in such graphs is the detection of communities. Informally, communities are collections of nodes that are more connected to each other than to nodes outside the community. One way to model communities is by introducing objective functions reflecting the quality of a partition of the set of nodes. The problem then reduces to an optimisation problem finding the best partition given an objective function. An interesting extension is when the network is evolving as reflected by an evolution in the community structure. Although evolution could in principle discontinuously impact the community structure, this is mostly not the case in reality nor what one wants in the mathematical models. Systems relying on community structure (think of an advisory system for public relations support) would probably

not be helped by erratically changing the viewed community structure. This adds a second criterion to the optimisation problem, namely that of not changing abruptly. [7, 8]. We will refer to this problem as the Dynamic Community Problem (DCP).

DCP has been studied as a weighted optimisation problem with two goal functions called snapshot cost ( $SC$ ) reflecting the absolute quality of the present structure and the temporal cost ( $TC$ ) reflecting the amount of change [17]:

$$cost = \alpha.SC + (1 - \alpha).TC$$

The solvers based on this cost function typically depend on a rather large number of parameters and decisions. From an application point of view, an appropriate measure for the community quality  $SC$  is selected from a large number of possibilities from clustering theory. The evolution in the structure must be properly modeled in  $TC$  and the parameter  $\alpha$  needs to be determined. These add to the parameters of the optimisation algorithm.

DCP is in essence situated in data science, the data being the interactions between the nodes of which the graph is a representation. When an optimisation approach is selected, parameter tuning, algorithm configuration and construction as discussed in Sect. 2.1 are needed.

As an example, we now discuss [12] in some detail. In this paper, DCP is recast as a multi-objective optimisation problem effectively getting rid of the weight  $\alpha$ . Four versions of the algorithm are investigated, each using a different definition for  $SC$ .  $TC$  is computed through an expression from information theory measuring the similarity between community partitions. The algorithm is based on NGS-II [10], and is taken from a *Matlab* library. The remaining parameters in the algorithm are linked to the genetic algorithm and the representation. Although the authors mention the possibility to tune these automatically, a trial and error approach is selected.

The experiments are performed on a number of synthetic data sets and on two data sets taken from real world cases. These data sets allow comparing against other algorithms, and it is shown that the algorithm outperforms existing ones. The sizes of the resulting graph range between  $10^2$  and  $10^3$  and must thus be considered small. The study is very similar to classical optimisation studies. An interesting analysis concerns complexity. The authors show that the complexity of the algorithm can be expressed as a function of the number of generations, the size of the population and the dimensions of the graph as

$$O(g.p.\log(p) \times (n.\log(n) + m))$$

where  $g$  is the number of generations,  $p$  is the size of the population,  $n$  is the number of nodes and  $m$  is the number of edges in the graph. Of course, the number of generations as well as the population size needed for acceptable results will vary with the size of the graph. The authors study the scalability of the method experimentally for numbers of nodes ranging from 128 to 4096 and numbers of edges ranging from 2938 to 65256. This allows quantifying these relationships to some extent. The authors conclude that a tradeoff between computing time and error gap is necessary.

So although the  $n \cdot \log(n)$  dependency is very promising and should allow handling much larger graphs, the hidden dependency in the required number of iterations and population size together with the rather large constant required by the genetic algorithm are limiting factors.

The example discussed illustrates how a standard optimisation approach could be used to solve a problem from data science. The low complexity of metaheuristic approaches holds a promise for obtaining good solutions for large graphs.

## 4 Conclusions and Future Work

Data science supporting optimisation was illustrated through an example of algorithm construction 2.1 and an attempt to support algorithm design 2.2.

The construction case is the most general case of algorithm configuration and tuning which are generalisations of algorithm selection. The example relied on a framework for the design of a multi objective algorithm and specialised in selecting its components automatically. The link with data science is inherent in real world problems. Even when using exact algorithms, the parameter setting strongly influences computation time. When heuristics are used, it is even more important as the heuristics are often known to work well on a limited set of problem instances only. A representative sample of the expected set is the starting point for automated algorithm construction.

Another link was highlighted when discussing automated support for algorithm design. Again a framework was selected, represented by a local search approach. The main design issue is the selection of appropriate neighbourhoods relying on experience and intuition of designers. The support comes in when trial versions of the algorithm are run using the neighbourhoods. By observing this process, characteristic of the neighbourhoods emerged, suggesting good and worse sets. The example was taken one step further when the neighbourhoods were clustered automatically and the algorithm tuner was shown to profit from this information by producing better parameter settings.

Optimisation for data science was discussed in Sect. 3.1. The example given was a community detection algorithm for large graphs. By modelling the problem as a multi-objective optimisation problem, an off-the-shelf genetic algorithm for multi-objective optimisation could be used. The implementation in *Matlab* outperformed existing methods. An interesting complexity analysis revealed good asymptotic behaviour if one does not take the size of the population into account.

These examples suggest a number of possibilities for future research. Template based algorithm construction is an example of how human knowledge and data can be combined in a methodology that semi-automatically produces optimisation algorithms. The human knowledge is expressed through the template and the creation of components. The data is taken into account by the automatic configuration tool *iracet* that essentially runs the algorithm against a representative set of problem instances. Similar ideas are used in hyperheuristics [5] where human knowledge and

intuition are introduced through so called low level heuristics which are set to use by a sophisticated optimisation algorithm, often a genetic algorithm. A challenge for these lines of research is formalisation of the specification. As the example illustrates, and as can be seen in hyperheuristics literature, specification of components or low level heuristics still requires high level expertise. A domain expert cannot be expected to master the kind of sophistication needed to write efficient code for those algorithm components. A formal definition or a declarative language for specifying this information could be an important step forward to take this methodology to practice [11].

The design support for analysing the neighbourhoods in the local search algorithm could speed up the task of specialist algorithm designers. The sound mathematical basis used in the realisation of the characteristics and in the clustering exercise have presently been implemented in R [9]. Concerning analysing the behaviour of neighbourhoods, an interesting follow up research could be about interaction between neighbourhoods as exemplified by [22].

The optimisation for data science example concentrates on a genetic algorithm. The complexity of the algorithm depends on the number of generations and the size of the population. The accuracy of the result depends on the same parameters so that a best compromise needs to be sought. It may be interesting to investigate a local search approach which often leads to more efficient coding schemes, requires less involved data structures and takes more advantage of delta evaluation of the goal function. The question of the asymptotic behaviour needs an answer when looking at larger graphs.

Finally, all these examples were inspired by problems from practice. In the history of optimisation, many insights have emerged from theoretical considerations. Less theoretical effort has been spent on algorithm selection, configuration or construction. We are thinking of a hypothetical setting where the instance distribution is described analytically. Questions could be on a specific algorithm template, e.g. for a traveling salesperson problem when the cities live on a given structure or on the asymptotic behaviour of a community detection algorithm in terms of execution time as well as solution quality.

These and other questions could play a similar role as the complexity studies on classic optimisation problems and could help us understand the issues of DFO and OFD better.

## References

1. Fermat-Torricelli problem. [https://www.encyclopediaofmath.org/index.php/Fermat-Torricelli\\_problem](https://www.encyclopediaofmath.org/index.php/Fermat-Torricelli_problem). Accessed 20 May 2017
2. Bezerra, L.C., López-Ibáñez, M., Stützle, T.: Automatic component-wise design of multiobjective evolutionary algorithms. *IEEE Trans. Evol. Comput.* **20**(3), 403–417 (2016)
3. Bleuler, S., Laumanns, M., Thiele, L., Zitzler, E.: PISAA platform and programming language independent interface for search algorithms. In: *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 494–508. Springer, Berlin Heidelberg (2003)



4. Brazil, M., Graham, R.L., Thomas, D.A., Zachariasen, M.: On the history of the euclidean steiner tree problem. *Archi. Hist. Exact Sci.* **68**(3), 327–354 (2014)
5. Burke, E.K., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., Qu, R. Hyper-heuristics: a survey of the state of the art. *J. Op. Res. Soc.* (2013) (to appear)
6. Burke, E.K., Hyde, M.R., Kendall, G., Ochoa, G., Ozcan, E., Woodward, J.R.: Exploring hyper-heuristic methodologies with genetic programming. In: *Computational Intelligence*, pp. 177–201. Springer, Berlin Heidelberg (2009)
7. Chakrabarti, D., Kumar, R., Tomkins, A.: Evolutionary clustering. In: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '06*, pp. 554–560. ACM, New York, NY, USA (2006)
8. Chi, Y., Song, X., Zhou, D., Hino, K., Tseng, B.L.: On evolutionary spectral clustering. *ACM Trans. Knowl. Discov. Data*, **3**(4), 17:1–17:30, December (2009)
9. Dang, N.T.T., De Causmaecker, P.: Characterization of neighborhood behaviours in a multi-neighborhood local search algorithm. In: *International Conference on Learning and Intelligent Optimization*, pp. 234–239. Springer International Publishing (2016)
10. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.A.M.T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Trans. Evol. Comput.* **6**(2), 182–197 (2002)
11. Devriendt, J., De Causmaecker, P., Denecker, M.: Transforming constraint programs to input for local search. In: *The Fourteenth International Workshop on Constraint Modelling and Reformulation*, pp. 1–16 (2015)
12. Folino, F., Pizzuti, C.: An evolutionary multiobjective approach for community discovery in dynamic networks. *IEEE Trans. Knowl. Data Eng.* **26**(8), 1838–1852 (2014)
13. Fortunato, S.: Community detection in graphs. *Phys. Rep.* **486**(3), 75–174 (2010)
14. Fortunato, S., Hric, D.: Community detection in networks: a user guide. *Phys. Rep.* **659**, 1–44 (2016)
15. Hales, T., Adams, M., Bauer, G., Dang, T.D., Harrison, J., Le Truong, H., Kaliszky, C., Magron, V., McLaughlin, S., Nguyen, T.T., Nguyen, Q.T., Nipkow, T., Obua, S., Pleso, J., Rute, J., Solovyyev, A., Hoai Thi Ta, A., Tran, T.N., Thi Trieu, D., Urban, J., Khac Vu, K., Zumkeller, R.: A formal proof of the Kepler conjecture. *ArXiv e-prints*, January (2015)
16. Hey, T., Tansley, S., Tolle, K.M., et al.: The fourth paradigm: data-intensive scientific discovery (2009)
17. Kim, M.-S., Han, J.: A particle-and-density based evolutionary clustering method for dynamic networks. *Proc. VLDB Endow.* **2**(1), 622–633 (2009)
18. Krasnogor, N., Smith, J.: A tutorial for competent memetic algorithms: model, taxonomy, and design issues. *IEEE Trans. Evol. Comput.* **9**(5), 474–488 (2005)
19. Liefvooghe, A., Jourdan, L., Talbi, E.-G.: A software framework based on a conceptual unified model for evolutionary multiobjective optimization: Paradiseo-moeo. *Eur. J. Op. Res.* **209**(2), 104–112 (2011)
20. López-Ibáñez, M., Dubois-Lacoste, J., Cáceres, L.P., Birattari, M., Stützle, T.: Iterated racing for automatic algorithm configuration. The irace package. *Op. Res. Persp.* **3**, 43–58 (2016)
21. Mascia, F., López-Ibáñez, M., Dubois-Lacoste, J., Stützle, T.: Grammar-based generation of stochastic local search heuristics through automatic algorithm configuration tools. *Comput. Op. Res.* **51**, 190–199 (2014)
22. Misir, M., Verbeeck, K., De Causmaecker, P., Berghe, G.V.: An intelligent hyper-heuristic framework for chesc 2011. In: *Learning and Intelligent Optimization*, pp. 461–466. Springer, Berlin Heidelberg (2012)
23. Schrijver, A.: On the history of combinatorial optimization (till 1960). *Handb. Op. Res. Manag. Sci.* **12**, 1–68 (2005)
24. Srour, A., De Causmaecker, P.: Towards automatic design of adaptive evolutionary algorithms (2017)

# From Mixed-Integer Linear to Mixed-Integer Bilevel Linear Programming

Matteo Fischetti

**Abstract** Bilevel Optimization is a very challenging framework where two players (with different objectives) compete for the definition of the final solution. In this paper we address a generic mixed-integer bilevel linear program, i.e., a bilevel optimization problem where the objective functions and constraints are all linear, and some variables are required to take integer values. We briefly describe some main ingredients of a branch-and-cut general-purpose framework for the exact solution (under appropriate assumptions) of mixed-integer bilevel linear programs.

## 1 Introduction

Consider a general bilevel optimization problem of the form

$$\min_{x \in \mathbb{R}^{n_1}, y \in \mathbb{R}^{n_2}} F(x, y) \quad (1)$$

$$G(x, y) \leq 0 \quad (2)$$

$$y \in \arg \min_{y' \in \mathbb{R}^{n_2}} \{f(x, y') : g(x, y') \leq 0\}, \quad (3)$$

where  $F, f : \mathbb{R}^{n_1+n_2} \rightarrow \mathbb{R}$ ,  $G : \mathbb{R}^{n_1+n_2} \rightarrow \mathbb{R}^{m_1}$ , and  $g : \mathbb{R}^{n_1+n_2} \rightarrow \mathbb{R}^{m_2}$ . Let  $n = n_1 + n_2$  denote the total number of decision variables.  $F(x, y)$  and  $G(x, y) \leq 0$  denote the *leader* objective function and constraints, respectively, while (3) defines the so-called *follower* subproblem. In case of multiple optimal solutions of the follower subproblem, we assume that one with minimum leader cost among those with  $G(x, y) \leq 0$  is chosen—i.e., we consider the *optimistic* version of bilevel optimization.

It is well known that one can define the follower value function for a given  $x \in \mathbb{R}^{n_1}$

$$\Phi(x) = \min_{y \in \mathbb{R}^{n_2}} \{f(x, y) : g(x, y) \leq 0\}, \quad (4)$$

---

M. Fischetti (✉)  
DEI, University of Padova, Padua, Italy  
e-mail: matteo.fischetti@unipd.it

and then restate the bilevel optimization problem as follows:

$$\min F(x, y) \tag{5}$$

$$G(x, y) \leq 0 \tag{6}$$

$$g(x, y) \leq 0 \tag{7}$$

$$(x, y) \in \mathbb{R}^n \tag{8}$$

$$f(x, y) \leq \Phi(x). \tag{9}$$

The above optimization problem is very hard (both theoretically and in practice) even if one assumes convexity of  $F, G, f$  and  $g$  (which would imply that of  $\Phi$ ), due to the intrinsic nonconvexity of (9). For example, the NP-hard 0-1 Integer Linear Program

$$\min c^T x \tag{10}$$

$$Ax = b, \quad x \in \{0, 1\}^n \tag{11}$$

can be rephrased as the following bilevel problem with linear objectives and constraints, and continuous  $x$  and  $y$  variables:

$$\min c^T x \tag{12}$$

$$Ax = b, \quad x \in [0, 1]^n \tag{13}$$

$$y = 0 \tag{14}$$

$$y \in \arg \min_{y'} \left\{ - \sum_{j=1}^n y'_j : y'_j \leq x_j, y'_j \leq 1 - x_j \quad \forall j = 1, \dots, n \right\} \tag{15}$$

where, for each  $j = 1, \dots, n$ , the follower variable  $y_j$  captures the fractional part of  $x_j$ , hence it must be zero in any integer leader solution  $x$  as stated in (14).

A point  $(x, y) \in \mathbb{R}^n$  is *bilevel infeasible* if it violates (9), while it is *bilevel feasible* if it satisfies all constraints (6)–(9).

By removing condition (9) one gets the the so-called *High Point Relaxation* (HPR). For the sake of simplicity, we assume that HPR is feasible and bounded, and that the minimization problem in (4) is bounded for every feasible solution of HPR.

In this paper we will focus on *Mixed-Integer Bilevel Linear Programs* (MIBLPs) where some/all variables are required to be integer, and all HPR constraints (plus objective functions) are linear/affine or stipulate the integrality of some variables. The first generic branch-and-bound approach to the MIBLPs has been given in [8], where the authors propose to solve HPR embedded into a branch-and-bound scheme. A sound branch-and-cut approach has been proposed in [3, 4], that cuts off integer bilevel infeasible solutions by adding cuts exploiting the integrality property of the leader and the follower variables. The authors have also provided an open-source MIBLP solver `MiBS` [9]. Other generic approaches for MIBLPs are the *branch-and-sandwich* method of [7], and the *watermelon* algorithm of [10].

We will next briefly outline a main ingredient of the branch-and-cut solver recently proposed in [5, 6], namely, the derivation of bilevel-specific intersection cuts [1].

## 2 Mixed-Integer Bilevel Linear Programming

Let the MIBLP of interest be stated as:

$$\min F(x, y) \quad (16)$$

$$G(x, y) \leq 0 \quad (17)$$

$$g(x, y) \leq 0 \quad (18)$$

$$(x, y) \in \mathbb{R}^n \quad (19)$$

$$f(x, y) \leq \Phi(x) \quad (20)$$

$$x_j \text{ integer, } \forall j \in J_1 \quad (21)$$

$$y_j \text{ integer, } \forall j \in J_2, \quad (22)$$

where  $F, G, f, g$  are now assumed to be affine functions, sets  $J_1 \subseteq \{1, \dots, n_1\}$  and  $J_2 \subseteq \{1, \dots, n_2\}$  identify the (possibly empty) indices of the integer-constrained variables in  $x$  and  $y$ , respectively, and the value function  $\Phi(x)$  reads

$$\Phi(x) = \min_{y \in \mathbb{R}^{n_2}} \{f(x, y) : g(x, y) \leq 0, y_j \in \mathbb{Z} \forall j \in J_2\}. \quad (23)$$

As already mentioned, dropping (20) leads to the HPR, which is a Mixed-Integer Linear Program (MILP) in our setting. Dropping integrality conditions as well leads to the LP relaxation of HPR, namely (16)–(19), a Linear Program (LP) which will be denoted by  $\overline{\text{HPR}}$ .

The approach of [5, 6] uses a standard simplex-based branch-and-cut algorithm to solve HPR, where the nonlinear constraint (20) is enforced, on the fly, by additional linear cuts. A minimal requisite for the correctness of such an approach is the ability of cutting any *vertex*, say  $(x^*, y^*)$ , of  $\overline{\text{HPR}}$  which happens to satisfy the integrality requirements (21) and (22) but is bilevel infeasible because

$$f(x^*, y^*) > \Phi(x^*), \quad (24)$$

thus preventing a wrong update of the incumbent solution of the branch-and-cut scheme.

### 3 Intersection Cuts for MIBLPs

The generation of Intersection Cuts (ICs) [1] that are able to cut a given infeasible HPR solution  $(x^*, y^*)$  requires the definition of two sets:

- (1) a *cone* pointed at  $(x^*, y^*)$  that contains all the bilevel feasible solutions, and
- (2) a *convex set*  $S^*$  that contains  $(x^*, y^*)$  but no bilevel feasible solutions in its interior.

As customary in mixed-integer programming, ICs are generated for vertices  $(x^*, y^*)$  of an LP relaxation of the problem to be solved, so a suitable cone is just the corner polyhedron associated with the corresponding optimal basis. All relevant information in this cone is readily available in the “optimal tableau” and requires no additional computational effort; see, e.g., [2] for details.

As to the convex set  $S^*$ , one can exploit the following result.

**Lemma 1** *For any  $\hat{y} \in \mathbb{R}^{n_2}$ , the set*

$$S(\hat{y}) = \{(x, y) \in \mathbb{R}^n : f(x, y) \geq f(x, \hat{y}), g(x, \hat{y}) \leq 0\} \quad (25)$$

*does not contain any bilevel feasible point in its interior.*

*Proof* It is enough to prove that no bilevel feasible  $(x, y)$  exists such that  $f(x, y) > f(x, \hat{y})$  and  $g(x, \hat{y}) < 0$ . We will in fact prove a tighter result where the latter condition is replaced by  $g(x, \hat{y}) \leq 0$ , as this will be required in the proof of the next theorem. Indeed, for any bilevel feasible solution  $(x, y)$  with  $g(x, \hat{y}) \leq 0$ , one has  $f(x, y) \leq \Phi(x) = \min_{y'} \{f(x, y') : g(x, y') \leq 0\} \leq f(x, \hat{y})$ .

In some relevant settings, the above result can be strengthened to obtain the following enlarged bilevel-free set.

**Theorem 1** *Assume that  $g(x, y)$  is integer for all HPR solutions  $(x, y)$ . Then, for any  $\hat{y} \in \mathbb{R}^{n_2}$ , the extended set*

$$S^+(\hat{y}) = \{(x, y) \in \mathbb{R}^n : f(x, y) \geq f(x, \hat{y}), g(x, \hat{y}) \leq 1\} \quad (26)$$

*does not contain any bilevel feasible point in its interior, where 1 denotes a vector of all ones.*

*Proof* To be in the interior of  $S^+(\hat{y})$ , a bilevel feasible  $(x, y)$  should satisfy  $f(x, y) > f(x, \hat{y})$  and  $g(x, \hat{y}) < 1$ . By assumption, the latter condition can be replaced by  $g(x, \hat{y}) \leq 0$ , hence the claim follows from the proof of previous lemma.

In the IC context, for a given  $(x^*, y^*)$  one can therefore define the convex set  $S^*$  required at item (2) above, as the set (25) or (26) with

$$\hat{y} = \arg \min_y \{f(x^*, y) : g(x^*, y) \leq 0, y_j \in \mathbb{Z} \forall j \in J_2\}. \quad (27)$$

Indeed, the resulting set  $S^*$  does not contain any bilevel feasible point in its interior, as required, while  $(x^*, y^*) \in S^*$  because of (24) and  $\Phi(x^*) = f(x^*, \hat{y})$  by definition. Note that  $\hat{y}$  is well defined when  $(x^*, y^*)$  is a solution of HPR, and that  $S^*$  is a convex polyhedron in the MIBLP case.

However, the generation of a *violated* IC also requires that  $(x^*, y^*)$  belongs to the *interior* of  $S^*$ , which can only be guaranteed for those MIBLPs for which Theorem 1 can be applied. This includes MIBLPs with *all-integer follower* where  $J_2 = \{1, \dots, n_2\}$ , all  $g$ -coefficients are integer, and for all  $x_j$ 's appearing with nonzero coefficient in some follower constraint are integer constrained.

*Example* Figure 1 illustrates the application of ICs on an example given in [8], which is frequently used in the literature:

$$\min_{x \in \mathbb{Z}} -x - 10y \quad (28)$$

$$y \in \arg \min_{y' \in \mathbb{Z}} \{ y' : \quad (29)$$

$$-25x + 20y' \leq 30 \quad (30)$$

$$x + 2y' \leq 10 \quad (31)$$

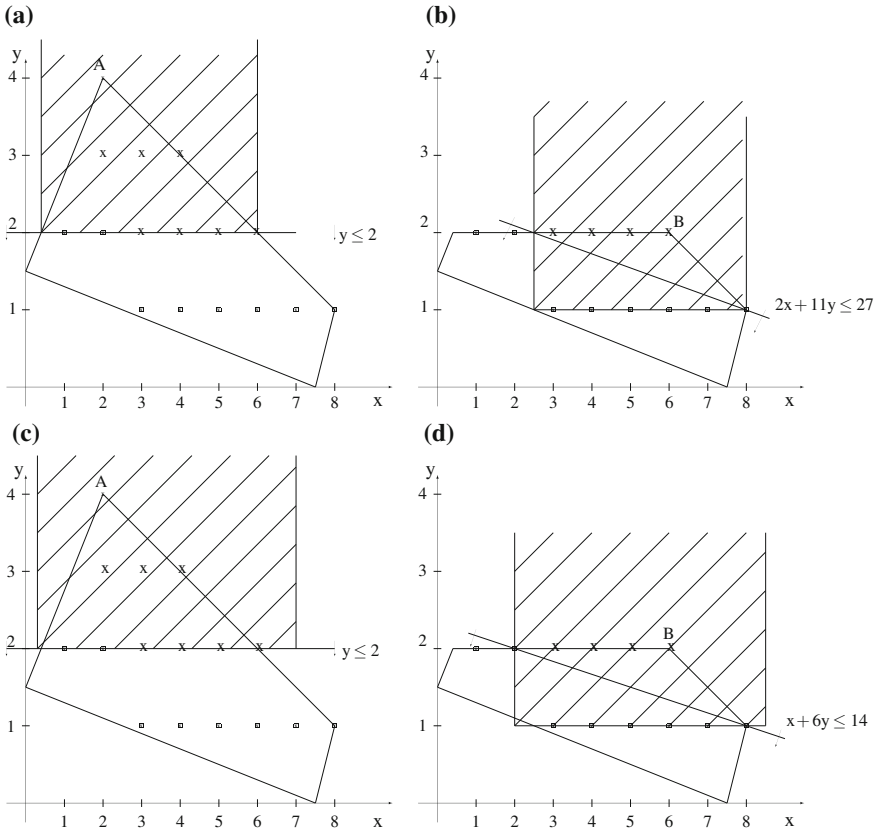
$$2x - y' \leq 15 \quad (32)$$

$$2x + 10y' \geq 15 \}. \quad (33)$$

In this all-integer example, there are 8 bilevel feasible points (depicted as crossed squares in Fig. 1), and the optimal bilevel solution is (2, 2). The drawn polytope corresponds to the HPR feasible set.

We first apply the definition of the bilevel-free set from Lemma 1 with  $\hat{y}$  defined as in (27). After solving the first HPR, the point  $A = (2, 4)$  is found. This point is bilevel infeasible, as for  $x^* = 2$  we have  $f(x^*, y^*) = y^* = 4$  while  $\Phi(x^*) = 2$ . From (27) we compute  $\hat{y} = 2$  and the intersection cut derived from the associated  $S(\hat{y})$  is depicted in Fig. 1a. In the next iteration, the optimal HPR solution moves to  $B = (6, 2)$ . Again, for  $x^* = 6$ ,  $f(x^*, y^*) = y^* = 2$  while  $\Phi(x^*) = 1$ . So we compute  $\hat{y} = 1$  and generate the IC induced by the associated  $S(\hat{y})$ , namely  $2x + 11y \leq 27$  (cf. Fig. 1b). In the next iteration, the fractional point  $C = (5/2, 2)$  is found and  $\hat{y} = 1$  is again computed. In this case,  $C$  is not in the interior of  $S(\hat{y})$  so we cannot generate an IC cut violated by  $C$ , and we proceed and optimize the current HPR to integrality by using a standard MILP solver. This produces the optimal HPR solution (2, 2) which happens to be bilevel feasible and hence optimal.

We next apply the definition of the enlarged bilevel-free set from Theorem 1 (whose assumption is fulfilled in this example) with  $\hat{y}$  defined again as in (27); see Fig. 1c and d. After the first iteration, the point  $A = (2, 4)$  is cut off by a slightly larger  $S^+(\hat{y} = 2)$ , but with the same IC as before ( $y \leq 2$ ). After the second iteration, from the bilevel infeasible point  $B = (6, 2)$  we derive a larger set  $S^+(\hat{y} = 1)$  and a stronger IC ( $x + 6y \leq 14$ ). In the third iteration, solution  $D = (2, 2)$  is found which is the optimal bilevel solution, so no branching is required in this example.



**Fig. 1** Illustration of the effect of alternative intersection cuts for a notorious example from [8]. Shaded regions correspond to the bilevel-free sets for which the cut is derived

### 4 Informed No-Good Cuts

A known drawback of ICs is their dependency on the LP basis associated with the point to cut, which can create cut accumulation in the LP relaxation and hence shallow cuts and numerical issues. Moreover, ICs are not directly applicable if the point to cut is not a vertex of a certain LP relaxation of the problem at hand, as it happens e.g. when it is computed by the internal MILP heuristics. We next describe a general-purpose variant of ICs whose derivation does not require any LP basis and is based on the well-known interpretation of ICs as disjunctive cuts. It turns out that the resulting inequality is valid and violated by any bilevel infeasible solution of HPR in the relevant special case where all  $x$  and  $y$  variables are binary.

Suppose we are given a point  $\xi^* = (x^*, y^*) \in \mathbb{R}^n$  and a polyhedron

$$S^* = \{ \xi \in \mathbb{R}^n : \alpha_i^T \xi \leq \alpha_{i0}, i = 1, \dots, k \}$$

whose interior contains  $\xi^*$  but no feasible points. Assume that variable-bound constraints  $l \leq \xi \leq u$  are present, where some entries of  $l$  or  $u$  can be  $-\infty$  or  $+\infty$ , respectively. Given  $\xi^*$ , let

$$L := \{j : \xi_j^* - l_j \leq u_j - \xi_j^*\} \text{ and } U := \{1, \dots, n\} \setminus L$$

and define the corresponding linear mapping  $\xi \mapsto \bar{\xi} \in \mathbb{R}^n$  with  $\bar{\xi}_j := \xi_j - l_j$  for  $j \in L$ , and  $\bar{\xi}_j := u_j - \xi_j$  for  $j \in U$  (variable shift and complement).

By assumption, any feasible point  $\xi$  must satisfy the disjunction

$$\bigvee_{i=1}^k \{ \xi \in \mathbb{R}^n : \sum_{j=1}^n \alpha_{ij} \xi_j \geq \alpha_{i0} \}, \quad (34)$$

whereas  $\xi^*$  violates all the above inequalities. Now, each term of (34) can be rewritten in terms of  $\bar{\xi}$  as

$$\sum_{j=1}^n \bar{\alpha}_{ij} \bar{\xi}_j \geq \bar{\beta}_i := \alpha_{i0} - \sum_{j \in L} \alpha_{ij} l_j - \sum_{j \in U} \alpha_{ij} u_j, \quad (35)$$

with  $\bar{\alpha}_{ij} := \alpha_{ij}$  if  $j \in L$ ,  $\bar{\alpha}_{ij} = -\alpha_{ij}$  otherwise.

If it happens that  $\bar{\beta}_i > 0$  for all  $i = 1, \dots, k$ , one can first normalize the above inequalities to the form  $\sum_{j=1}^n (\bar{\alpha}_{ij}/\bar{\beta}_i) \bar{\xi}_j \geq 1$ , then one can derive the valid disjunctive cut in the  $\bar{\xi}$  space

$$\sum_{j=1}^n \bar{\gamma}_j \bar{\xi}_j \geq 1, \quad (36)$$

where  $\bar{\gamma}_j := \max\{\bar{\alpha}_{ij}/\bar{\beta}_i : i = 1, \dots, k\}$ , and finally one can transform it back to the  $\xi$  space by substitution.

It can be seen that the above construction can always be applied in case  $\xi_j^* \in \{l_j, u_j\}$  for all  $j = 1, \dots, n$  (because  $\bar{\beta} > 0$ ), and produces a violated cut (as  $\bar{\xi}^* = 0$ ). In all other cases, the above cut separation is just heuristic.

The inequality (36) is called *Informed No-Good* (ING) cut in [5], in that it can be viewed as a strengthening of the no-good cut  $\sum_{j \in L} \xi_j + \sum_{j \in U} (1 - \xi_j) \geq 1$  often used for bilevel problems with all-binary variables—and in many other constraint programming and mathematical optimization contexts. Indeed, standard no-good cuts correspond to the “almost blind” choice  $S^* = \{\xi \in \mathbb{R}^n : \xi_j \leq 1 \forall j \in L, 1 - \xi_j \leq 1 \forall j \in U\}$ .

**Acknowledgements** This research was partially supported by MiUR, Italy (PRIN2015 project) and by the Vienna Science and Technology Fund (WWTF) through project ICT15-014.



## References

1. Balas, E.: Intersection cuts—a new type of cutting planes for integer programming. *Op. Res.* **19**(1), 19–39 (1971)
2. Conforti, M., Cornuejols, G., Zambelli, G.: *Integer Programming*. Springer International Publishing (2014)
3. DeNegre, S.: Interdiction and discrete bilevel linear programming. In: PhD Thesis. Lehigh University (2011)
4. DeNegre, S., Ralphs, T.K.: A branch-and-cut algorithm for integer bilevel linear programs. In: *Operations Research and Cyber-Infrastructure*, pp. 65–78. Springer (2009)
5. Fischetti, M., Ljubić, I., Monaci, M., Sinnl, M.: Intersection cuts for bilevel optimization. In: Louveaux, Q., Skutella, M. (eds.) *Integer Programming and Combinatorial Optimization: 18th International Conference, IPCO 2016, Liège, Belgium, June 1–3, 2016, Proceedings*, Springer International Publishing, pp. 77–88. (2016) (Extended version mathematical programming)
6. Fischetti, M., Ljubić, I., Monaci, M., Sinnl, M.: A new general-purpose algorithm for mixed-integer bilevel linear programs. *Operations Research*
7. Kleniati, P.-M., Adjiman, C.S.: A generalization of the branch-and-sandwich algorithm: from continuous to mixed-integer nonlinear bilevel problems. *Comput. Chem. Eng.* **72**, 373–386 (2015)
8. Moore, J., Bard, J.: The mixed integer linear bilevel programming problem. *Op. Res.* **38**(5), 911–921 (1990)
9. Ralphs, T.K.: MibS. <https://github.com/TKralphs/MibS>
10. Xu, P., Wang, L.: An exact algorithm for the bilevel mixed integer linear programming problem under three simplifying assumptions. *Comput. Op. Res.* **41**, 309–318 (2014)

# **Part II**

## **Data Science**

# Outperforming Image Segmentation by Exploiting Approximate K-Means Algorithms

Flora Amato, Mario Barbareschi, Giovanni Cozzolino,  
Antonino Mazzeo, Nicola Mazzocca and Antonio Tammaro

**Abstract** Recently emerged as an effective approach, Approximate Computing introduces a new design paradigm for trade system overhead off for result quality. Indeed, by relaxing the need for a fully precise outcome, Approximate Computing techniques allow to gain performance parameters, such as computational time or area of integrated circuits, by executing inexact operations. In this work, we propose an approximate version of the K-means algorithm to be used for the image segmentation, with the aim to reduce the area needed to synthesize it on a hardware target. In particular, we detail the methodology to find approximate variants of the K-means and some experimental evidences as a proof-of-concept.

## 1 Introduction

Nowadays, images are one of the most important medium for conveying information. It is not only related to multimedia applications, but rather, even thanks to a larger view offered by Big Data problems, understanding images and extracting the

---

F. Amato · M. Barbareschi · G. Cozzolino (✉) · A. Mazzeo ·  
N. Mazzocca · A. Tammaro

Dipartimento di Ingegneria Elettrica e delle Tecnologie Dell'informazione,  
University of Naples Federico II, Via Claudio, 21 - 80125 Naples, Italy  
e-mail: giovanni.cozzolino@unina.it

F. Amato  
e-mail: flora.amato@unina.it

M. Barbareschi  
e-mail: mario.barbareschi@unina.it

A. Mazzeo  
e-mail: mazzeo@unina.it

N. Mazzocca  
e-mail: nicola.mazzocca@unina.it

A. Tammaro  
e-mail: a.tammaro@zoho.com

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_4

information from them is still a challenging task. An example of the use of images is for biomedical applications, such as medical diagnosis, or navigation of robots.

This paper focuses on a particular image processing task, which is the segmentation. Mainly, the image segmentation involves a clustering algorithm, that is a process that partitions a group of data points into a small number of sets, namely the clusters. As many Big Data applications are based on image clustering [1, 2], new research challenges, related to system performance in executing such a task, are arising.

Implementing algorithms as hardware accelerator could deeply affect system performance, for instance involving Field Programmable Gate Array (FPGA) devices [3, 4]: for instance, Hussain et al. in [5] proposed a pipelined technique for implementing a hardware accelerator hosted by a Xilinx Virtex 4 FPGA device. Compared to a software implementation, they claimed to reach a speed-up of about 51 and a gain of 200 as energy saving. Contrary to the architecture proposed in [5], which employed a range analysis for fixed-point format, we started from another concept, which is given by the Approximate Computing (AC).

Indeed, researchers investigated on the benefits given by relaxing the need of having a correct result in output employing incorrect operations [6]. First, there are applications which tolerate a certain amount of error because are inherently robust to input/computational error. This property is named inherent resiliency application and opens the opportunity of performing sub-operations in an approximate manner, saving computational resources, in case of software execution, or, conversely, hardware area/energy.

Bearing in mind these considerations, in this paper we apply AC to obtain an approximate hardware acceleration of the K-means algorithm. In particular, we exploit the IDEAS software for exploring variants of the K-means by applying the bit-width reduction technique together with a multi-objective optimization approach. We report a preliminary experimental campaign in which we show the efficacy of the bit-width reduction technique combined with a hardware implementation of the K-means. Moreover, by implementing two opposite solutions, namely one with the highest error and highest performance, on a Xilinx Zynq-7020 FPGA, we show the actual hardware overhead reduction.

The remainder of the paper is structured as follows: Sect. 2 gives a brief overview about related work, while Sect. 3 details the K-means algorithm and the methodological flow of IDEAS; in Sect. 4 we report preliminary experimental results while Sect. 5 draws the conclusions.

## 2 Approximate Computing in the Literature

The inadequacy of current technological and architectural methodologies is making AC very popular in the scientific literature [7, 8]. Indeed, it has been proved that algorithms do not need to be exactly executed, but rather they output good-enough results despite some of their inner operations are executed in an approximate manner [6].

This fact provides a new design axis which allows to trade output quality off for performance gain. In case of software AC techniques, performance is meant to be execution time, or memory accesses, while in case of hardware techniques, area and energy consumption of the integrated circuits are taken into account.

As previously stated, algorithms which benefit from the AC are characterized by what is defined as inherent application resiliency [6]. Most of them come from domains in which outputs have to be interpreted directly by human end-users, such as signal processing or multimedia, even though the inherent application resiliency has been proved for search algorithm, machine learning and pattern recognition [9–11].

AC techniques involve any stack of a digital system, including: synchronization elision [12], lossy compression [13], loop perforation [14] for the software layer; unreliable DRAM [15], approximate processing [16], bit-width reduction [17] for hardware and memory layers.

In order to manage AC techniques and system variants obtained by employing them, the design process needs for automatic tools. Most of tools proposed in the literature work on a specific design-level entry and by considering few AC techniques [18].

In [19], authors proposed a new paradigm based on code mutation for implementing a generic AC design exploration, named  $\mathbb{I}DE\mathbb{A}$  ( $\mathbb{I}DE\mathbb{A}$  is a Design Exploration tool for Approximate Algorithms). As discussed in Sect. 3.1, we exploit  $\mathbb{I}DE\mathbb{A}$  for approximating the K-means algorithm.

### 3 Image Clustering Through Approximate K-Means

The K-means is one of the most popular algorithm employed for clustering applications. Indeed, K-means contemplates  $k$  centroids that are used to define clusters.

To comprehend how it performs, let us consider  $n$  data points  $\underline{x}_i, i = 1 \dots n$  that have to be partitioned in  $k$  clusters, such that each point in a given cluster is characterized by the nearest mean with others. Hence, K-means aims to find the positions  $\mu_i, i = 1 \dots k$  of the clusters that minimize the distance between each data point in the cluster. Formally, K-means solve the following problem:

$$\arg \min_c \sum_{i=1}^k \sum_{x \in c_i} d(x, \mu_i) = \arg \min_c \sum_{i=1}^k \sum_{x \in c_i} \|x - \mu_i\|_2^2$$

where  $c_i$  is the set of points that belong to cluster  $i$ .

A typical K-means clustering implementation makes use of the square Euclidean distance:

$$d(x, \mu_i) = \|x - \mu_i\|_2^2.$$

A special case of K-means algorithm was originally designed by Lloyd [20] and it is used to solve the K-means clustering problem.

The Algorithm 1 describes the Lloyd's proposal: it eventually converges to a point, although it is not necessarily the minimum of the sum of squares. That is because the problem is non-convex and converges to a local minimum. Indeed, the algorithm stops when the assignments do not change from one iteration to the next one.

**Data:** Given  $K$ , the number of clusters, and  $|c|$ = number of elements in  $c$   
Initialize the center of the clusters  $\mu_i = \text{some value}$ ,  $i = 1, \dots, k$ ;  
**while** *convergence is not reached* **do**  
    Attribute the closest cluster to each data point  
     $c_i = \{j : d(x_j, \mu_i) \leq d(x_j, \mu_l), l \neq i, j = 1, \dots, n\}$ ;  
    Set the position of each cluster to the mean of all data points belonging to that cluster  $\mu_i$   
     $= \frac{1}{|c_i|} \sum_{j \in c_i} x_j, \forall i$ ;  
**end**

### **Algorithm 1:** K-means Algorithm

In this paper, we use K-means in order to segment a set of images. The algorithm performs first an initialization phase, and then proceeds with the segmentation of the image by grouping pixels in clusters.

## **3.1 Approximate Computing by Mutating Algorithms**

Even though the observation behind the AC is intuitive, a more spread adoption of AC techniques is hampered by design challenges. First, the choice of a specific approximate technique is not trivial since, as previously debated, it affects a specific part of a computing system and there are no tools which are able to directly evaluate the impact of approximating an algorithm. Moreover, the lack of both a general methodology and a tool which makes it automatic, hinders AC to be integrated with common hardware or software design flows.

Authors in [19] introduced a novel approach which promises to be general and extensible. Indeed, they fostered a mutation-based technique to generate approximate variants by employing arbitrary AC techniques by means of software definition. Indeed, the algorithm to approximate is provided in form of C/C++ project, while AC techniques are implemented by extending clang-Chimera, a mutation tool based on the clang/LLVM compiler suite. The mutation is preparatory to an exploration phase, which is performed by tuning each applied AC technique on the code. It follows that the cardinality of the solutions space is directly related to the number of configuration for each technique and for every mutated point of the code. The exploration of such a space is driven by a branch and bound (B&B) approach which prune subtrees that crosses a given error/quality threshold according to a user-defined metric.

As highlighted in [21], the number of variants that need to be explored could make the approach unfeasible. Moreover, another drawback is the limitation given by the error or quality function, which does not take into account the performance gaining of variants.

Taking into account these considerations, rather than a B&B algorithm, in this paper we exploit a multi-objective optimization approach by means of meta-heuristic searching algorithms. Indeed, each AC technique can be characterized in terms of error by executing the mutated and configured C/C++ algorithm but it needs also to be tagged about the performance gain that it guarantees. As for the former, the end-user can identify a suited error function which estimates, by means of a specific metric, the distance of results between the approximate variant. As for the latter, there are different performance functions that the end-user has to take into account. They depends on the kind of AC technique since they affect different layers of a computing system.

For our experimental set-up, we exploit the bit-width reduction, which is a technique that uses fewer bits to represent variables involved into an algorithm, compared with the original type. The technique can be applied by considering standard representation, meant to be executed in software [22], or custom for hardware implementation [17].

## 4 Preliminary Experimental Campaign

In order to exploit the IDEA approach exploring the solution space by means of a multi-objective optimization approach, we developed a tool which considers two objective functions: (i) error and (ii) reward. Contrary to the the B&B, which only considers the error in each exploration step, we propose to evaluate each approximate variant with a further characterization which takes into account the performance gain. While the error function has to be minimized, we want a reward as high as possible.

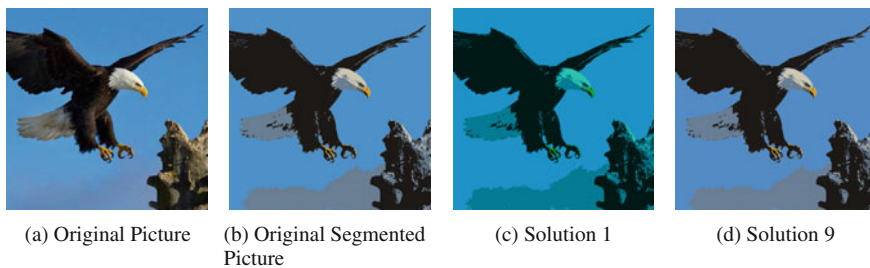
Thus, we coded a K-means algorithm in C/C++ and ran clang-Chimera to get a mutated code. We set-up the error function as the mean square error (MSE) calculated between 20 original segmented pictures and the approximate ones. As for the reward function, we considered the number of bits saved for addition and subtraction operations and the squared number of bits saved for multiplication and division ones.

For brevity sake, we report only result related to the usage of the Non-dominated Sorting Genetic Algorithm (NGSA-II), even though we applied other meta-heuristic algorithms which were provided by the ParadisEO framework suite [23]. In our set-up, we configured NGSA-II to work with an initial population of 500 approximate variants and 1000 generations.

Preliminary experimental evidences are reported in Table 1, which contains 9 approximate configurations output as final population from the NGSA-II algorithm. As one can notice, the chosen reward and the MSE are Inverse proportionality and the highest reward value is associated with an MSE equal to 0.1.

**Table 1** Final population get from NGS-II over approximate configuration of K-means

Solution	MSE	Reward
1	0.101754	1084
2	0.0996137	1054
3	0.0954545	1040
4	0.0954545	1011
5	0.0903658	977
6	0.0601272	950
7	0.0584772	391
8	0.0583554	387
9	0.043019	366

**Fig. 1** Example of an image segmentation process performed with the original K-means and with approximate variants found by IDEAS**Table 2** Hardware resources saving of solution 1 and 9 w.r.t. fully-precise K-means algorithm

Solution	LUTs (%)	Registers (%)	Slices (%)
1	45	80	56
9	87	93	91

Examples of how approximate K-means variants perform are given in Fig. 1. In particular, we reported the original picture (Fig. 1a) and the original segmented version (Fig. 1b) against the one obtained from solution 1 and 9 of the Table 1 (Fig. 1c and 1d respectively).

We further implemented solution 1 and 9 on a Xilinx Zynq-7020 FPGA in order to evaluate the area saving. Table 2 illustrates the area saving in % of both the solutions w.r.t. a fully-precise K-means implementation. The approximate solution 1 yield an area saving in terms of look-up tables (LUTs) of about half a quantity.



## 5 Conclusion and Final Discussion

Approximate Computing is claimed to be a very effective methodology to tackle with tight application constraints. The basic idea of outperforming a specific task just renouncing to perfect quality results can be used by a plethora of applications, especially ones related to image elaboration. In this work we proposed and applied a methodology to approximate the K-means algorithm by exploiting of the bit-width reduction technique. We proved the feasibility of approximation by mutation, introduced by the  $\mathbb{I}D\mathbb{E}\mathbb{A}$  methodology, in combination with a multi-objective optimization approach. We ran some preliminary experimental runs that highlight the suitability of the proposed approach. In particular, the most significant approximate configuration introduced an MSE of 0.1 yield to about 50% of area saving w.r.t. the original K-means algorithm implemented over a Xilinx Zynq-7020 device.

As part of future work, we aim to explore different multi-objective optimization algorithms, as well as to try different reward metrics in order to investigate their ability to find better approximate solutions.

## References

1. Thilagamani, S., Shanthi, N.: A survey on image segmentation through clustering. *Int. J. Res. Rev. Inf. Sci.* **1**(1), 14–17 (2011)
2. Chen, M., Mao, S., Liu, Y.: Big data: a survey. *Mob. Netw. Appl.* **19**(2), 171–209 (2014)
3. Cilardo, A.: New techniques and tools for application-dependent testing of FPGA-based components. *IEEE Trans. Ind. Inform.* **11**(1), 94–103 (2015)
4. Cilardo, A., Fusella, E., Gallo, L., Mazzeo, A.: Automated synthesis of FPGA-based heterogeneous interconnect topologies. In: 2013 23rd International Conference on Field Programmable Logic and Applications (FPL), pp. 1–8. IEEE (2013)
5. Hussain, H.M., Benkrid, K., Seker, H., Erdogan A.T.: FPGA implementation of K-means algorithm for bioinformatics application: an accelerated approach to clustering microarray data. In: 2011 NASA/ESA Conference on Adaptive Hardware and Systems (AHS), pp. 248–255. IEEE (2011)
6. Chippa, V.K., Chakradhar, S.T., Roy, K., Raghunathan, A.: Analysis and characterization of inherent application resilience for approximate computing. In: Proceedings of the 50th Annual Design Automation Conference, p. 113. ACM (2013)
7. Bosio, A., Virazel, A., Girard, P., Barbareschi, M.: Approximate computing: design and test for integrated circuits. In: 2017 8th Latin American Test Symposium (LATS), p. 1–6. IEEE (April 2016)
8. Bosio, A., Debaud, P., Girard, P., Guilhot, S., Valka, M., Virazel, A.: Auto-adaptive ultra-low power IC. In: 2016 International Conference on Design and Technology of Integrated Systems in Nanoscale Era (DTIS), pp. 1–6 (April 2016)
9. Venkataramani, S., Chakradhar, S.T., Roy, K., Raghunathan, A.: Approximate computing and the quest for computing efficiency. In: Proceedings of the 52nd Annual Design Automation Conference, p. 120. ACM (2015)
10. Amato, F., Barbareschi, M., Casola, V., Mazzeo, A.: An FPGA-based smart classifier for decision support systems. In: Intelligent Distributed Computing VII, pp. 289–299. Springer (2014)
11. Amato, F., Mazzeo, A., Moscato, V., Picariello, A.: A framework for semantic interoperability over the cloud. In: 2013 27th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 1259–1264. IEEE (2013)

12. Misailovic, S., Sidiroglou, S., Rinard, M.C.: Dancing with uncertainty. In: Proceedings of the 2012 ACM Workshop on Relaxing Synchronization for Multicore and Manycore Scalability, pp. 51–60. ACM (2012)
13. Samadi, M., Lee, J., Jamshidi, D.A., Hormati, A., Mahlke, S.: Sage: self-tuning approximation for graphics engines. In: Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture, pp. 13–24. ACM (2013)
14. Sidiroglou-Douskos, S., Misailovic, S., Hoffmann, H., Rinard, M.: Managing performance vs. accuracy trade-offs with loop perforation. In: Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering, pp. 124–134. ACM (2011)
15. Liu, S., Pattabiraman, K., Moscibroda, T., Zorn, B.G.: Flicker: saving refresh-power in mobile devices through critical data partitioning. In: Proceedings of International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS). Citeseer (2009)
16. Yetim, Y., Martonosi, M., Malik, S.: Extracting useful computation from error-prone processors for streaming applications. In: Design, Automation and Test in Europe Conference and Exhibition (DATE), 2013, pp. 202–207. IEEE (2013)
17. Barbareschi, M., Iannucci, F., Mazzeo, A.: Automatic design space exploration of approximate algorithms for big data applications. In: 2016 30th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 40–45. IEEE (2016)
18. Mittal, S.: A survey of techniques for approximate computing. *ACM Comput. Surv. (CSUR)* **48**(4), 62 (2016)
19. Barbareschi, M., Iannucci, F., Mazzeo, A.: An extendible design exploration tool for supporting approximate computing techniques. In: 2016 International Conference on Design and Technology of Integrated Systems in Nanoscale Era (DTIS), pp. 1–6. IEEE (2016)
20. Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R., Wu, A.Y.: An efficient K-means clustering algorithm: analysis and implementation. *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(7), 881–892 (2002)
21. Barbareschi, M., Iannucci, F., Mazzeo, A.: A pruning technique for B&B based design exploration of approximate computing variants. In: 2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pp. 707–712. IEEE (2016)
22. Rubio-González, C., Nguyen, C., Nguyen, H.D., Demmel, J., Kahan, W., Sen, K., Bailey, D.H., Iancu, C., Hough, D.: Precimonious: tuning assistant for floating-point precision. In: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, p. 27. ACM (2013)
23. Liefvooghe, A., Jourdan, L., Legrand, T., Humeau, J., Talbi, E.G.: Paradiseo-moeo: a software framework for evolutionary multi-objective optimization. In: Advances in Multi-Objective Nature Inspired Computing, pp. 87–117. Springer (2010)

# Approximate Decision Tree-Based Multiple Classifier Systems

Mario Barbareschi, Cristina Papa and Carlo Sansone

**Abstract** Implementing hardware accelerators of multiple classifier systems assures an improving in performance: on one hand, the combination of multiple classifiers outcomes is able to improve classification accuracy, with respect to a single classifier; on the other hand, implementing the prediction algorithm by means of an integrated circuit enables classifier systems with higher throughput and better latency compared with a pure software architecture. Although, this approach requires a very high amount of hardware resources, limiting the adoption of commercial configurable devices, such as Field Programmable Gate Arrays. In this paper, we exploit the application of Approximate Computing to trade classification accuracy off for hardware resources occupation. Specifically, we adopt the bit-width reduction technique on a multiple classifier system based on the Random Forest approach. A case study demonstrates the feasibility of the methodology, showing an area reduction ranging between 8.3 and 72.3%.

## 1 Introduction

Today, digital data growth is a spreading phenomenon that involves all kinds of computing systems, spanning from wearable devices to cloud computing, and it is commonly identified by the Big Data term. It is not just about the data size or data volume, but rather it is the way whereby the classic architectural approaches handle information elaboration. Such inadequacy is generating main challenges for the so-called Big Data problems and it is tight related to the technological limitations and applications constraints.

---

M. Barbareschi · C. Papa · C. Sansone (✉)

DIETI - University of Naples Federico II, via Claudio 21, 80125 Naples, Italy

e-mail: carlosan@unina.it

M. Barbareschi

e-mail: mario.barbareschi@unina.it

C. Papa

e-mail: cri.papa@studenti.unina.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,

DOI 10.1007/978-3-319-67308-0\_5

Researchers put a significant effort for data classification, machine learning and pattern recognition algorithms, since they have to cope with large datasets and with an important number of data samples per time unit, as well as the tight response time required to process each sample. As for the former point, since even a negligible percentage of misclassifications will eventually impact on a relevant number of samples, new learning algorithms have been developed, with the aim of improving the classification accuracy. As for the latter points, which are namely throughput and latency, proposed solutions make use of novel hardware architectures [10, 11].

Authors of [4] demonstrated the feasibility to implement Multiple Classifier Systems (MCSs), based on the Decision Tree (DT) model, onto Field Programmable Gate Array (FPGA) devices. First, DTs are one of the most suited classification models for being implemented in hardware, since the visiting algorithm does not involve expensive arithmetic computations, thus it solely requires comparisons. Moreover, the literature proved the efficacy of MCSs based on DTs in many Pattern Recognition applications, characterized by a large pattern variability [12].

Due to finite availability of hardware resources, some MCS model configurations do not fit inside a target device and requires more resources for being synthesized. For this reason, in this paper, we introduce the Approximate Computing methodology which adds design axes that take into account the trade off between classification accuracy and area overhead. In particular, through a case study, we prove that MCSs can be successfully approximated introducing a negligible error by means of the bit-width reduction technique.

The remainder of the paper is structured as follows: Sect. 2 gives a brief research related work overview; Sect. 3 illustrates the methodological flow to approximate a multiple DT model, while in Sect. 4 we show the experimental results; Sect. 5 draws the conclusion.

## 2 Related Work

As previously discussed, DT models are appropriate for being realized as hardware primitives. Several approaches are possible: they have different architectural characteristics, i.e. sequential synthesis [3] or fully-parallel [2], different application fields, such as health monitoring systems [7], traffic analysis [15], security purposes [6], supporting single classifiers or multiple classifier systems [17].

As for the architecture, the design of a sequential or parallel accelerator impacts the latency of the circuits as well as the area/energy, since a parallel architecture would lead to bigger and faster circuits compared to the sequential ones [7]. Those parameters have to be taken into account accordingly to their target applications in order to fit design constraints. As for the MCSs, they deal with the performance of a single classification model that cannot be optimized beyond a certain threshold despite efforts at either refining the model description or the classification. Indeed, adopting a MCS together with a combining function compensates the conditioning of a single classifier [19].

The Approximate Computing term has been introduced for indicating a design paradigm that implements efficient hardware circuits and software components by tolerating inexact outputs. Researchers proved the efficacy of imprecise computing for efficient design of software/hardware modules that implement approximate algorithms thanks to their inherent resiliency, that is the property of some algorithms to return acceptable outcomes despite some of its inner computations being approximate [9, 18].

In this sense, the Approximate Computing design exploits approaches that leverage inherent resiliency through optimizations which trade outputs quality off for better performance, such as time, energy consumption, occupied area, and so on [8]. The inherent resiliency is prevalent in data analytics, web search and wireless communications, which exhibits the same behavior of applications whose output is intended to be interpreted by humans [14].

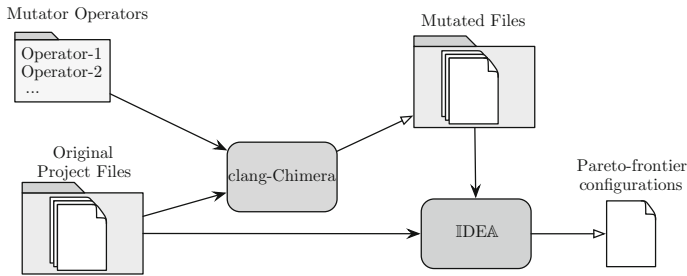
### 3 Approximate Computing for DT Models

As briefly discussed in the previous section, MCSs exploit a combining function to decide for a single output starting from multiple model outcomes. Such function opens the opportunity to approximate each model of the MCS, since errors introduced turn out compensated. Thus, it is clear that the prediction algorithm of a MCS is characterized by a strong inherent resiliency that can be exploited to enhance performance.

In order to provide best approximate configurations of MCSs, in terms of performance gain and introduced accuracy error, we foster (i) the approximation of bit-width reduction of features involved in the model and (ii) the automatic exploration of approximate configurations for different MCSs.

To this aim, we start from a trained model, described through a Predictive Model Markup Language (PMML) file, an XML-based format developed by Data Mining Group (DMG), intended to support different statistical and data mining models [13]. Then, we obtain a C++ project for handle PMML models by means of CodeSynthesis XSD framework that performs statically-typed C++ bindings from XML Schema definitions. The next step is the developing of a C++ MCS predictor in order to generate approximate variants w.r.t. a given Approximate Computing technique. As described in the next subsection, we employ clang-Chimera and IDE $\mathbb{A}$  tools to obtain Pareto-optimum approximate configurations.

The last operation is the generation of corresponding approximate hardware predictors. To this aim, we branch the PMML2VHDL project [1] for the PMML2AVHDL (PMML to Approximate VHDL). This tool is able to automatically generate an approximate VHDL project implementing the trained model from a PMML, configured by means of the IDE $\mathbb{A}$  output.



**Fig. 1** Execution flow of IDEAS tool

### 3.1 The IDEAS Work-Flow

The methodology behind the IDEAS tool, first introduced in [5], is shortly illustrated in Fig. 1. The process considers algorithms coded in C/C++ language, hence the original algorithm, as well as approximate variants obtained from it, can be easily simulated in software.

The entire process involved 2 tools, namely clang-Chimera and IDEAS, as draw in Fig. 1. The former is a mutation engine for C/C++ code based on the Clang compiler and written in C++. It applies a list of configured mutators on the code that implement a specific Approximate Computing Technique. The latter is a design exploration tool which performs a branch and bound (B&B) search. In particular, it applies a greater approximation for each mutation in the code till to cross a given error threshold. It follows that, at each iteration, IDEAS generates an approximate version of the target algorithm, which exhibits an error greater than or equal to the error that has been observed in previous iterations. Once the exploration terminates, the leaves of the execution tree are Pareto-optimum configurations of the original algorithm, such that they cannot be further approximated without crossing the chosen error threshold.

It is worth noting that the whole work-flow is generic and can be extended to implement different Approximate Computing techniques and different approximation campaigns, changing the error/quality function and the threshold. Indeed, as for the approximation of predictor algorithm, we extended IDEAS to support an operator which reduces the bit-width of numeric types, while the quality (and, hence, the error) is evaluated by comparing the accuracy of the approximate model w.r.t. the original one.

## 4 Experimental Results

The approach introduced in this paper is evaluated through a spam email detection case study. Classifying emails as SPAM or no-SPAM is a problem characterized

by a significant amount of information to be classified. We consider the *Spambase* dataset, freely available on UCI Machine Learning repository [16]. In particular, it contains 4601 instances, including 1813 SPAM emails, characterized by 57 features, expressed in *real-number* format, that indicate whether a particular word or character was frequently occurring in the email, and one binary class that denotes if the emails are SPAM or no-SPAM. We adopt KNIME Analytics Platform for training a DT classifier, and different Random Forest classifiers made up of 10, 20, 30 and 40 trees. We split the dataset into training and test sets considering a 50/50 holdout.

Among different Approximate Computing techniques, we target the bit-width reduction approach for representing model features and, as previously stated, we extended  $\mathbb{I}DE\mathbb{A}$  accordingly. Reducing the bit-width leads to represent values neglecting some of the least-significant bits, while the remaining bits assume the same weight of the previous representation. The impact of such technique on a hardware implementation turns out to reduce the size of the circuits, in particular, neglecting gates involved in the comparators.

In order to evaluate the error introduced by approximations,  $\mathbb{I}DE\mathbb{A}$  performs accuracy evaluation by searching for all possible features bit-reduction combinations with a B&B technique. In particular,  $\mathbb{I}DE\mathbb{A}$  approximates the value of the mantissa. This means that the whole solutions space amounts to  $52^{57}$ , where 52 are the bits of the mantissa (since KNIME adopts IEEE 754 double precision for real numbers), and 57 the number of features. We further configure  $\mathbb{I}DE\mathbb{A}$  with a maximum error threshold of 0.1%, that is a low error value on the classification accuracy, and with an error function that determines the absolute differences between the original and the approximate model accuracy.

The number of solutions that we need to explore is clearly not feasible: even if each single run of  $\mathbb{I}DE\mathbb{A}$  took 1 ns to complete, the time required to evaluate 1% of that solutions would take about  $2 \times 10^{79}$  years. Hence, we adopted two strategies to reduce the huge amount of possible combinations: (i) pre-pruning of the B&B nodes choosing as tree root an approximate variant in which we already remove some bits from mantissae; (ii) grouping features by information gain.

As for the former purpose, we conducted preliminary tests considering all the features approximated to the same number of bits and by setting the error threshold to zero. Hence, for each classifier, we obtained the starting bit-width reduction searching root. It is worth noting that, contrary to KNIME, we used the IEEE 754 single precision since it does not introduce any error in the trained predictor.

As for the latter point, we design the following tests by dividing the features into 2, 4, 6 and 8 groups. In each of these, a group consists of the remaining features approximate to the same number of bits. So, for example, the test with 8 groups is constituted by the seven most discriminative features (according to the information gain criterion), individually approximated, and by the group of the remaining ones.

Table 1 reports result in terms of occupied hardware resources, namely number of look-up tables, registers and slices, collected at post place-and-route phase, w.r.t. the Xilinx Virtex 5 XC5VLX110T. The first row of each experimental campaign reports result of the original model, while the others report two different tests. Indeed,

**Table 1** Hardware resources for different implementations of Spambase predictor models on the Xilinx Virtex 5

#Trees	Accuracy	Feature groups	Saved bits	Error	LUT	Register	Slices		
1	0.907	N/D	N/D	N/D	1198	2302	925		
		1	969	4.346e-04	644	1478	583		
			855	0	644	1478	583		
		2	969	4.346e-04	644	1478	583		
			967	0	657	1482	560		
		4	982	8.692e-04	631	1454	606		
			871	0	727	1595	680		
		6	988	4.346e-04	632	1442	538		
			880	0	727	1577	654		
		8	998	8.692e-04	630	1423	580		
			1004	0	635	1421	455		
		10	0.939	N/D	N/D	N/D	15330	10704	6607
				1	1083	8.692e-04	9815	8499	3535
					855	0	11853	8999	4163
2	1086			8.692e-04	9784	8488	3486		
	862			0	11785	8983	3015		
4	1090			8.692e-04	9740	8472	2653		
	979			0	10341	8745	3685		
6	1096			8.692e-04	9714	8460	3504		
	989			0	10310	8724	3580		
8	1101			8.692e-04	9646	8430	3571		
	998			0	10232	8696	3643		
20	0.944			N/D	N/D	N/D	26384	18280	9580
				1	1083	8.692e-04	18809	15528	6231
					855	0	22372	16113	8044
		2	1086	8.692e-04	18720	15502	6062		
			862	0	22222	16075	7266		
		4	1091	8.692e-04	18661	15482	7379		
			873	0	22122	16051	8451		
		6	1096	8.692e-04	18623	15467	7277		
			1036	0	19544	15723	7370		
		8	1102	0	18091	15233	8341		
			1102	0	18091	15233	8341		
		30	0.945	N/D	N/D	N/D	37345	25444	10706
				1	1140	8.692e-04	26066	21798	8121
					1026	0	29260	22721	8341
2	1142			8.692e-04	26031	21785	7393		
	1027			0	29258	22719	8212		
4	1200			4.346e-04	23423	20795	6678		
	1031			0	28976	22629	8491		

(continued)



**Table 1** (continued)

#Trees	Accuracy	Feature groups	Saved bits	Error	LUT	Register	Slices		
		6	1202	4.346e-04	23330	20758	6710		
			1200	0	23433	20797	6661		
		8	1204	4.346e-04	23249	20707	6610		
			1202	0	23357	29746	6624		
		40	0.948	N/D	N/D	N/D	47455	32370	13671
				1	1083	8.692e-04	37055	29218	10611
					1026	0	38840	29682	10735
				2	1086	8.692e-04	36925	29171	10237
1030	0				38637	29633	10883		
4	1046			8.692e-04	34451	28409	10196		
	1037			0	38593	29613	10870		
6	1149			4.346e-04	34381	28385	10164		
	1146			0	34460	28412	10233		
8	1203			8.692e-04	30847	27034	9060		
	1152			0	33811	28170	10017		

for each feature group test, we pick Pareto-optimum solutions: the most bit-saving solution and one with minimum error.

At first glance, the experiments highlight the effectiveness of the approach, i.e. area saving, ranging from 15.21 to 47.41% LUTs, from 8.30 to 38.27% Registers, from 11.78 to 59.85% Slices. Furthermore, in the majority of cases, the 8 groups tests saves more resources. This confirms that the best results are obtained by separately approximating the most discriminative features, while leaving the other in a single group.

As demonstrated in Table 1, the percentage of saved area occupation decreases as the number of trees increases. This is because the voter is not involved by the approximation process.

Moreover, Pareto frontiers are characterized by solutions which are not affected by any classification accuracy error: for each experimental campaign, there is a Pareto-solution with no error. This is the most important result because it demonstrates that for MCSs model this approach allows us to reduce hardware circuitry even without compromising model accuracy. Let us consider the comparison between a single DT and the Random Forest model made up of 10 trees. The DT model occupies about 5% of the Virtex5 resources. The original Random Forest with 10 trees increases the accuracy of 3.2% and the area it occupies is about 40%. In our tests we demonstrate that 40% of the Virtex5 area can be occupied by approximating without errors a Random Forest made up of 30 trees, so giving rise to a further improvement of the accuracy (up to the 4%).

## 5 Conclusions

In this paper, we advanced the implementation of hardware circuits for Decision Tree-based Multiple Classifier Systems by exploiting FPGA devices. Compared to previous approaches, we adopted the Approximate Computing bit-width reduction technique to reduce the area overhead despite the introduction of classification accuracy loss. To this aim, we illustrated a novel Approximate Computing approach, which makes use of a mutation-code tool and a design space exploration for searching Pareto-optimum solutions, that is clang-Chimera and IDEA tools. We extended these tools and proposed an original design flow to obtain an approximated variant of Decision Tree-based Multiple Classifier Systems models.

Through a real case study, we proved the feasibility of the proposed methodology, which was always able to produce approximated models with no classification errors, saving up to the 72.31% of the occupied resources.

## References

1. Amato, F., Barbareschi, M., Casola, V., Mazzeo, A., Romano, S.: Towards automatic generation of hardware classifiers, pp. 125–132. Springer International Publishing, Cham (2013)
2. Barbareschi, M.: Implementing hardware decision tree prediction: a scalable approach. In: 2016 30th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 87–92. IEEE (2016)
3. Barbareschi, M., Battista, E., Mazzocca, N., Venkatesan, S.: A hardware accelerator for data classification within the sensing infrastructure. In: 2014 IEEE 15th International Conference on Information Reuse and Integration (IRI), pp. 400–405. IEEE (2014)
4. Barbareschi, M., Del Prete, S., Gargiulo, F., Mazzeo, A., Sansone, C.: Decision tree-based multiple classifier systems: an FPGA perspective. In: International Workshop on Multiple Classifier Systems, pp. 194–205. Springer (2015)
5. Barbareschi, M., Iannucci, F., Mazzeo, A.: An extendible design exploration tool for supporting approximate computing techniques. In: 2016 International Conference on Design and Technology of Integrated Systems in Nanoscale Era (DTIS), pp. 1–6. IEEE (2016)
6. Barbareschi, M., Mazzeo, A., Miranda, S.: Adopting decision tree based policy enforcement mechanism to protect reconfigurable devices. In: Intelligent Interactive Multimedia Systems and Services 2016, pp. 73–81. Springer International Publishing (2016)
7. Barbareschi, M., Romano, S., Mazzeo, A.: A cloud based architecture for massive sensor data analysis in health monitoring systems. In: 2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), pp. 521–526. IEEE (2015)
8. Chakradhar, S.T., Raghunathan, A.: Best-effort computing: re-thinking parallel software and hardware. In: Proceedings of the 47th Design Automation Conference, pp. 865–870. ACM (2010)
9. Chippa, V.K., Chakradhar, S.T., Roy, K., Raghunathan, A.: Analysis and characterization of inherent application resilience for approximate computing. In: Proceedings of the 50th Annual Design Automation Conference, p. 113. ACM (2013)
10. Cilaro, A.: New techniques and tools for application-dependent testing of FPGA-based components. *IEEE Trans. Ind. Inform.* **11**(1), 94–103 (2015)
11. Cilaro, A., Fusella, E., Gallo, L., Mazzeo, A.: Automated synthesis of fpga-based heterogeneous interconnect topologies. In: 2013 23rd International Conference on Field Programmable Logic and Applications (FPL), pp. 1–8. IEEE (2013)

12. Gargiulo, F., Mazzariello, C., Sansone, C.: Multiple classifier systems: theory, applications and tools. In: *Handbook on Neural Information Processing*, pp. 335–378. Springer (2013)
13. Guazzelli, A., Lin, W.C., Jena, T.: PMML in action: unleashing the power of open standards for data mining and predictive analytics. CreateSpace (2012)
14. Han, J., Orshansky, M.: Approximate computing: an emerging paradigm for energy-efficient design. In: *2013 18th IEEE European Test Symposium (ETS)*, pp. 1–6. IEEE (2013)
15. Jiang, W., Prasanna, V.K.: Scalable packet classification on FPGA. *IEEE Trans. Very Large Scale Integration (VLSI) Syst.* **20**(9), 1668–1680 (2012)
16. Lichman, M.: UCI machine learning repository (2013)
17. Struharik, R.: Decision tree ensemble hardware accelerators for embedded applications. In: *2015 IEEE 13th International Symposium on Intelligent Systems and Informatics (SISY)*, pp. 101–106. IEEE (2015)
18. Venkataramani, S., Chakradhar, S.T., Roy, K., Raghunathan, A.: Approximate computing and the quest for computing efficiency. In: *Proceedings of the 52nd Annual Design Automation Conference*, p. 120. ACM (2015)
19. Xu, L., Krzyzak, A., Suen, C.Y.: Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Trans. Syst. Man Cybern.* **22**(3), 418–435 (1992)

# Look-Up Tables for Efficient Non-Linear Parameters Estimation

Stefano Marrone, Gabriele Piantadosi, Mario Sansone  
and Carlo Sansone

**Abstract** In the Big-Data era, many engineering tasks have to deal with extracting valuable information from large amount of data. This is supported by different methodologies, many of which strongly rely on curve fitting (both linear and non-linear). One of the most common approach to solve this kind of problems is the use of least squares method, usually by iterative procedures that can cause slowness when applied to problems that require to repeat the fitting procedure many times. In this work we propose a method to speed-up the curve fitting evaluation by means of a Look-up Table (LuT) approach, exploiting problems resilience. The considered case study is the fitting of breast Dynamic Contrast Enhanced-Magnetic Resonance Imaging (DCE-MRI) data to a pharmacokinetic model, that needs to be fast for clinical usage. To validate the proposed approach, we compared our results with those obtained by using the well-known Levenberg-Marquardt algorithm (LMA). Results show that the proposed approach and LMA are not statistically different in terms of MSE (with respect to the observed data) and in terms of Sum of Differences in the parameter and in the solution spaces. Considering the computational effort, the proposed LuT approach is an order of magnitude faster than the LMA on the same dataset.

**Keywords** Curve fitting · Least squares · Levenberg-Marquardt · Look-up tables  
DCE-MRI · Pharmacokinetic models · PBPK

---

S. Marrone · G. Piantadosi · M. Sansone · C. Sansone (✉)  
DIETI - University of Naples Federico II, via Claudio 21, 80125 Naples, Italy  
e-mail: carlosan@unina.it

S. Marrone  
e-mail: stefano.marrone@unina.it

G. Piantadosi  
e-mail: gabriele.piantadosi@unina.it

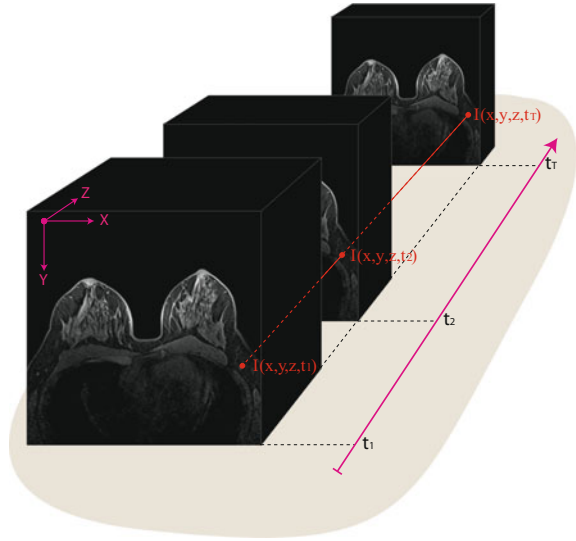
M. Sansone  
e-mail: msansone@unina.it

# 1 Introduction

Nowadays almost every field of research deals with extracting valuable information behind the huge amount of available raw data. Although this is supported by different methodologies, many of them strongly rely on curve fitting, an optimization method that aims to find the parameters' values of a mathematical model that best fit some given data points [1]. It has a crucial role in many engineering tasks, since the derived parameters allow the user to infer the function values for those points whose measurements are not available [2]. Common real-world problems must have constraints on parameters, since they often represent well-known natural or artificial properties (such as physics, biological, mechanical, etc.) that have to range within well defined boundaries. For this reason, it is common to have more equations than unknowns function parameters, so giving rise to an overdetermined system problem. The standard solving approach is the least squares method which consists in minimize the sum of squared residuals between the observed data (points to be fitted) and the fitted values, obtained by evaluating the fitting function [3]. Least-squares can be distinguished in linear and non-linear according to the linearity of residual in all the unknowns. This characterisation has a deep impact both on the solution evaluation feasibility and on the accuracy. The former has in fact a closed-form solution, while the latter usually does not and needs to be solved by different approaches, such as iterative procedures that operate by approximating step-by-step a linear system. Since many real engineering problems fall in non-linear least squares family, different procedures to deal with it were so far proposed. Among them, it is worth to mention the Levenberg-Marquardt algorithm (LMA) [4, 5], an iterative approach that merges Gauss-Newton algorithm and the Gradient Descent method to derive a more robust procedure (meaning that it can converge even starting far aside from the minimum). For its iterative nature, one of the main disadvantages of this algorithm is its slowness when applied to problems that require repeating the fitting procedure over a huge amount of different observed data. Despite this limitation, many engineering problems are resilient to small perturbations in the data, allowing to settle on a near-optimum solution. In this paper we proposed a method to speed-up the curve fitting parameter evaluation by means of a Look-up Table (LuT), properly built off-line. We will demonstrate that our approach can strongly speed-up the curve fitting, without affecting the reliability of the obtained results. One of the research field that deals with curve fitting (but that has also some resiliency capacity) is the Dynamic Contrast Enhanced-Magnetic Resonance Imaging (DCE-MRI, Fig. 1) automatic processing. In recent years it has gained popularity as an important complementary diagnostic methodology for early cancer detection [6, 7].

In DCE-MR images each voxel (volumetric pixel) is associated with a Time Intensity Curve (TIC), that reflects the absorption and the release of the contrast agent. A very important TIC property is that it can be modelled by means of different Physiologically-Based Pharmacokinetic (PBPK) models, leading the basis to accurately estimate the contrast agent concentration from the TIC data [8]. As a case study, we will focus on fitting the contrast agent concentration to a given

**Fig. 1** The DCE-MRI study is composed of different series acquired before and after the intravenous injection of a contrast agent



physiological model for breast DCE-MRI, with the aim to compare different motion correction techniques. To validate the suitability of the proposed approach, we compare our results with those obtained by using the Levenberg-Marquardt algorithm. The paper is organized as follows. In Sect. 2 we describe the proposed approach and present the considered case-study. In Sect. 3 we report the results of our analysis. Finally, in Sect. 4 we discuss the results and draw some conclusions.

## 2 The Proposed Approach

Extracting information behind the data implies defining the mathematical model that describes relations and processes behind the data itself. If the values of the model parameters are unknown, it is possible to estimate them from the observed data, through curve-fitting procedures. A non-linear least-squares approach usually requires an iterative approximation of the solution and, therefore, it is time expensive and computationally heavy. Our proposal aims to strongly reduce the computational overhead due to iterations, by approximating the solution using a one-time and off-line generated Look-Up Table (LuT), properly indexed to allow fast access to the required information (model parameters).

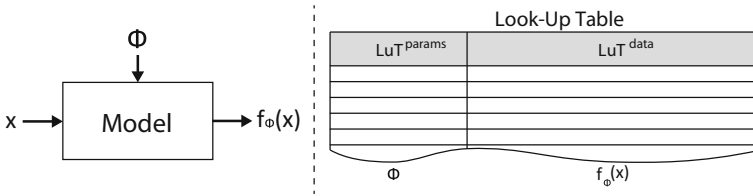


Fig. 2 Creating a LuT starting from a model

## 2.1 Curve Fitting Using Look-Up Tables

A LuT is, basically, a table composed of  $N$  rows  $R_i$ , each one containing a given parameters combination  $LuT_i^{params}$  and the values  $LuT_i^{data}$  of the function  $f_\phi(x)$  evaluated by using the parameters  $\phi$  in the same row (see Fig. 2).

### 2.1.1 Creating the LuT

Since the LuT represents the parameters and the relative function values for a specific problem, it can be calculated off-line and used when a parameter estimation is required. The procedure for creating a LuT requires to define the range  $[P_j^{min}, P_j^{max}]$  where each parameter  $P_j$  is defined and a step value  $\delta_j$  for sampling the values of these parameters. This range can be directly derived from the model: if the parameters represents known properties, their values can be constrained in a well-known range; otherwise, if the parameters are mathematical quantities, their range can be sharply chosen. Choosing the right  $\delta_j$  is crucial: while a too big sampling step can introduce an unacceptable parameter approximation, a too small sampling step will produce a bigger LuT.

### 2.1.2 Using the LuT

Obtaining the desired fitting parameters using the LuT requires to search in the table for the fitted values that better approximate the observed data and to extract the relative parameters. In practice, we are searching for the row  $R_i$  having the smallest difference between the observed data and the  $LuT_i^{data}$  in the least-square sense. This means that using the LuT for the fitting first  $i$  requires the evaluation of the differences along the whole table and then  $ii$  a search for their minimum. Due to the huge amount of data contained in the LuT, the time performance required by a sequential search could be comparable, or even worst, with respect to that of an iterative curve-fitting procedure. Moreover, since these differences depend on the observed data, it is not possible to evaluate them at creation-time, preventing the definition of a suitable index. To overcome this limit, we propose to pre-select a fraction of the LuT, enabling a sequential search on a smaller table. We chose to characterize





$$\begin{aligned}
C_v(t) &= K^{\text{trans}} \int_0^t e^{-\frac{K^{\text{trans}}}{v_e}(t-s)} \cdot C_p(s) ds + v_p C_p(t) \\
C_p(t) &= \sum_{i=1}^2 \frac{A_i}{\sigma_i \sqrt{2\pi}} e^{-\frac{(t-T_i)^2}{2\sigma_i^2}} + \alpha \frac{e^{-\beta t}}{1 + e^{-s(t-\tau)}}
\end{aligned} \tag{1}$$

where  $K^{\text{trans}}$  is the tissue volume transfer constant (or permeability) [ $\frac{1}{\text{min}}$ ];  $v_e$  is the EES volume per unit volume of tissue and  $v_p$  is the plasma volume fraction.

Parker et al. [11] report the ETK-P model parameters values obtained according to a population-averaged evaluation.

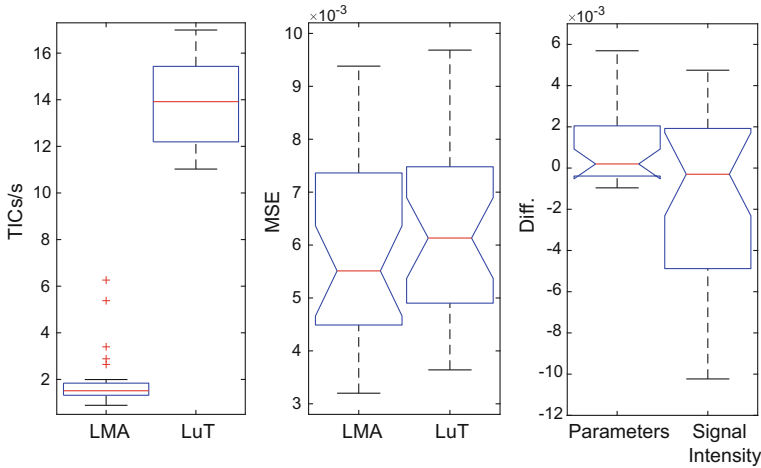
### 2.2.2 LuT in ETK-P Fitting

In the considered case-study we need to fit contrast agent concentration to the ETK-P model for each voxel within the breast volume. As shown in our previous works [9, 10], this operation is slow and can take up to 15 hours per patient. In this context, LuT has been created by using the following setting:  $K^{\text{trans}}$  ranges in  $[0, 10]$  with  $\delta = 0.01$ ;  $v_e$  ranges in  $(0, 1]$  with  $\delta = 0.01$ ;  $v_p$  ranges in  $(0, 1]$  with  $\delta = 0.01$ ;  $\gamma = 0.4$ ;  $K = 50$ .

The resulting LuT is composed exactly of 10.010.000 rows and 13 columns, where the first three are the parameters ( $LuT^{\text{params}}$ ), while the last ten are the values of the ETK-P model evaluated using the parameter in the same row ( $LuT^{\text{data}}$ ). At the same time, an array with the same number of rows is generated in order to contain the AUC index.

## 3 Experimental Results

In this work we used a dataset composed of breast DCE-MRI data from 30 patients (average age 40 years, in range 16–69) with 14 benign lesions and 16 malignant lesions, histopathologically proven. All patients underwent imaging with a 1.5 T scanner (Magnetom Symphony, Siemens Medical System, Erlangen, Germany) equipped with breast coil. DCE T1-weighted FLASH 3D coronal images (80 slices each) were acquired (Scanning Sequence/Variant: GR/SP; ScanOptions: PFP; RT/ET: 9.8/4.76 ms; FA: 25 degrees ; FoV  $370 \times 185 \text{ mm}^2$ ; thickness: 2 mm; gap: 0; acquisition time: 56s). One series ( $t_0$ ) was acquired before and 9 series ( $t_1$ – $t_9$ ) after the intravenous injection of Gadolinium-diethylene-triamine penta-acetic acid (Gd-DOTA, Dotarem, Guerbet, Roissy CdG Cedex, France). All the results are compared with those obtained by using the Levenberg-Marquardt algorithm (LMA) evaluated on an Intel Core i7–3630 QM 64 bit Quad Core 2, 4 GHz CPU equipped with 16 GB RAM. All comparisons are reported by using box-plots where median,



**Fig. 4** LuT performance evaluation. On the left: performance comparison in terms of throughput (TIC/s). In the middle: MSE comparison between LuT and LMA. On the right: the sum of differences in the parameter and curve spaces between LuT and LMA. With the aim of highlighting the medians and the corresponding IQRs, the MSE axis has been accordingly centred, causing some outliers to fall out of the plotted interval

quartiles, Inter-Quartile Ranges (IQRs) and outliers are shown. When needed, we have also drawn notches to highlight 95% confidence intervals of the medians.

Figure 4 (on the left) compares LuT and LMA throughput in terms of number of fitted TIC per second over the whole dataset. It is worth noticing that a fair comparison should also consider the LuT building time that, however, can be amortized over the total number of TIC to be fitted ( $\#fittings$ ) since the LuT has to be calculated only once. This implies a trade-off in preferring building the LuT versus the use of LMA, quantified by a break-even point  $N_b$ , representing how many successive fittings are required in order to make the LuT approach more convenient. In our case study, since  $N_b = 218400$  is less than  $\#fittings = 3207195$ , the LuT is preferable and its creation affects each fitting for less than 0.0094s.

Figure 4 (in the middle) compares LuT and LMA approaches in terms of goodness-of-fit (Mean Square Error - MSE) over the whole dataset. The MSE evaluated for both LMA and for proposed LuT approaches are comparable with 95% of confidence (as shown in the notches on the boxplot):  $MSE_{LMA} = 5.5119 \times 10^{-3} \pm 8.5270 \times 10^{-4}$  and  $MSE_{LuT} = 6.1329 \times 10^{-3} \pm 7.6470 \times 10^{-4}$ .

Finally, we compare the suitability of the proposed LuT approach with respect to the LMA by calculating the Sum of Differences both in the parameter and in the solution spaces over the whole dataset (Fig.4 — on the right). The boxplot also shows that the 0 value is contained within the 95% confidence interval for the sum of differences both in the parameters and in the solutions space (between LuT and LMA):  $DIFF_{parameters} = 2.0000 \times 10^{-4} \pm 7.2255 \times 10^{-4}$  and  $DIFF_{signal\_intensity} = -4.0000 \times 10^{-4} \pm 2.0180 \times 10^{-3}$ .

## 4 Discussion and Conclusions

The aim of this paper was to propose a method to speed-up the curve fitting problem by means of a Look-up Table (LuT) in those situations in which a fast near-optimum solution is strongly preferred over a more precise, but much slower, one. The considered case study is the fitting of breast DCE-MRI tumour data to a PBPK model. Results showed that the proposed LuT approach can be effectively used without affecting the reliability of the fitting results. This is supported by Fig. 4 (in the middle), in which it is possible to note that box plots of MSE evaluated between the fitted points and the original DCE-MRI data almost overlap. Results also show that the best fitting parameters obtained by applying the Levenberg-Marquardt are not statistically different from those obtained by applying the proposed LuT approach. The same can be stated for the fitted values. These assertions can be derived from Fig. 4 (on the right), showing that the 0 value is contained within the 95% confidence interval for the sum of differences both in the parameters and in the solutions space (between LuT and LMA). Finally, we compared the required fitting time. Figure 4 (on the left) shows that the proposed LuT approach is more than an order of magnitude faster than the LMA one in terms of number of fitted TIC per seconds. The speed-up is achieved since the LuT is evaluated off-line and only once for a given fitting function. This implies that, since the LuT evaluation effort is amortized over all patient, more time a given LuT is used, less its evaluation affects the computing time. Future works will investigate if it is worth applying the LuT approach even for ordinary least squares and when using others curve fitting resolution algorithms.

**Acknowledgements** The authors are grateful to Dr. Antonella Petrillo, Head of Division of Radiology, Department of Diagnostic Imaging, Radiant and Metabolic Therapy, “Istituto Nazionale dei Tumori Fondazione G. Pascale”-IRCCS, Naples, Italy, for providing access to DCE-MRI data. Moreover, we would like to thank Roberta Fusco, Ph.D., from the same institution, for the useful discussions.

## References

1. Arlinghaus, S.: Practical Handbook of Curve Fitting. CRC press (1994)
2. Hauser, J.R.: Numerical Methods for Nonlinear Engineering Models. Springer Science and Business Media (2009)
3. Lawson, C.L., Hanson, R.J.: Solving Least Squares Problems. SIAM (1995)
4. Levenberg, K.: A method for the solution of certain non-linear problems in least squares. *Quart. appl. math.* **2**(2), 164–168 (1944)
5. Marquardt, D.W.: An algorithm for least-squares estimation of nonlinear parameters. *J. soc. Ind. Appl. Math.* **11**(2), 431–441 (1963)
6. Marrone, S., Piantadosi, G., Fusco, R., Petrillo, A., Sansone, M., Sansone, C.: “Automatic lesion detection in breast DCE-MRI”. In: *Image Analysis and Processing (ICIAP)*, pp. 359–368. Springer, Berlin Heidelberg (2013)
7. Piantadosi, G., Marrone, S., Sansone, M., Sansone, C.: A secure scalable and versatile multi-layer clientserver architecture for remote intelligent data processing. *J. Reliab. Intell. Environ.* **1**, 173–187 (2015)

8. Schabel, M.C., Morrell, G.R., Oh, K.Y., Walczak, C.A., Barlow, R.B., Neumayer, L.A.: Pharmacokinetic mapping for lesion classification in dynamic breast MRI. *J. Magn. Reson. Imaging* **31**(6), 1371–1378 (2010)
9. Marrone, S., Piantadosi, G., Fusco, R., Petrillo, A., Sansone, M., Sansone, C.: “A Novel model-based measure for quality evaluation of image registration techniques in DCE-MRI”. In: *IEEE 27th 2014 CBMS*, pp. 209–214. IEEE (2014)
10. Piantadosi, G., Marrone, S., Fusco, R., Petrillo, A., Sansone, M., Sansone, C.: “Data-driven selection of motion correction techniques in breast DCE-MRI”. In: *IEEE International Symposium on Medical Measurements and Applications (MeMeA) Proceedings* (2015)
11. Parker, G.J.M., Roberts, C., Macdonald, A., Buonaccorsi, G.A., Cheung, S., Buckley, D.L., Jackson, A., Watson, Y., Davies, K., Jayson, G.C.: Experimentally-derived functional form for a population-averaged high-temporal-resolution arterial input function for dynamic contrast-enhanced MRI. *Magn. reson. med.* **56**, 993–1000 (2006)
12. Tofts, P.S., Brix, G., Buckley, D.L., Evelhoch, J.L., Henderson, E., Knopp, M.V., Larsson, H.B.W., Lee, T.-Y., Mayr, N.A., Parker, G.J.M.: “Estimating kinetic parameters from dynamic contrast-enhanced T<sub>1</sub>-weighted MRI of a diffusable tracer: standardized quantities and symbols”. *J. Magn. Reson. Imaging.* **10**(3), 223–232 (1999)

# On the UTA Methods for Solving the Model Selection Problem

Valentina Minnetti

**Abstract** In this paper, the multiple criteria UTA methods are proposed for solving the problem of model selection. The UTA methods realize a ranking of the models, from the best one to the worst one, by means of the comparisons of their global utility values. These values are computed by means of Linear Programming problems. Two UTA methods are illustrated. An example, that examines the performances of some classification models in the web context, is presented.

**Keywords** UTA methods · Linear programming · Classification models

## 1 Introduction

The model selection is one of the modern scientific enterprises [1]. It is an important part of any statistical analysis [1]. Both frequentist and Bayesian schools have dealt with this issue, proposing several algorithms for choosing a “good model” and proposing criteria for judging the quality of the models [1]. The Bayes factor, the Bayes Information Criterion (BIC), the Akaike Information Criterion (AIC), the bootstrap criterion, the cross-validation criterion, are some of the statistical criteria that evaluate the performances of the statistical models [1, 2]. As well as, the indices  $R^2$  and the Mallows’s statistic are used for the evaluation of the regression models [1, 2].

The model selection can be extended to all types of models, statistical and non statistical, whose performances can be measured by indicators and indices. When there are many models (models of different families or models in the same family), the choice of the best one can be a hard problem. So, rather than to choose the “best model”, according to some criterion, it is more reasonable to disregard those which are obviously poor, maintaining a subset for further consideration [1]. Also,

---

V. Minnetti (✉)

Faculty of Information Engineering, Department of Statistic Science,  
Informatics and Statistics, Sapienza University of Rome, Rome, Italy  
e-mail: valentina.minnetti@uniroma1.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_7

when there are many, different, conflicting criteria (to be minimized and/or to be maximized), the task of selecting a subset of models is a hard problem.

The proposal of this paper is to deal the problem of model selection with multiple criteria approach. In the context of Multiple Criteria Decision Aiding (MCDA), the UTA methods provide a solution to the problem of conflicting criteria, by creating a unique criterion that synthesizes all the criteria under consideration. By constructing a unique criterion, the selection of a subset of models is performed and thus, the deleting of the worst ones is a simple task.

In the next section, a brief introduction to MCDA and to two UTA methods are described. Then, the analysis of an example in the web context is presented.

## 2 The UTA and the UTASTAR Methods

In this section an introduction to the MCDA and to two UTA methods are presented, without too many details.

In MCDA, a finite set of  $n$  objects (alternatives, projects)  $A = \{a_1, a_2, \dots, a_n\}$  is evaluated by means of a finite set of  $m$  criteria  $G = \{g_1, g_2, \dots, g_m\}$ . A criterion is a real-valued function  $g_j : A \rightarrow \mathfrak{R}$ , whereby  $g_j(a_k)$  indicates the performance of the alternative  $a_k$  on the criterion  $g_j$ . The comparison of any pair of alternatives  $a_i$  and  $a_k$ , by means of a binary relationship, can be performed by comparing the values  $g_j(a_i)$  and  $g_j(a_k)$ , according to the structure of the criterion  $g_j$  [3]. For example, the true criterion works as follows:

- if  $g_j(a_t) = g_j(a_s)$ , then  $a_t I_j a_s$  or  $a_s I_j a_t$
- if  $g_j(a_t) > g_j(a_s)$ , then  $a_t P_j a_s$

where  $I_j$  represents the indifference binary relationship ( $\sim_j$ ) and  $P_j$  the strict preference binary relationship ( $\succ_j$ ) on the criterion  $g_j$ . Both binary relationships are transitive. Such structure of the preferences is rather rigid, because the DM must be rational. It means that a little difference of two performances is meaningful in order to express a strict preference binary relationship.

Let  $g_j^*$  and  $g_{j*}$  be the best and the worst values of the criterion  $g_j$ , respectively; let  $\underline{g}_j = \min_{a_k \in A} [g_j(a_k)]$  and  $\bar{g}_j = \max_{a_k \in A} [g_j(a_k)]$  be the minimum and the maximum value of the criterion  $g_j$ , respectively. A criterion  $g_j$  can be either of gain or cost type. In the first case, the Decision Maker (DM) prefers high values of  $g_j$  (i.e.  $g_j^* = \bar{g}_j$  or  $g_{j*} = \underline{g}_j$ ), while in the second case, the DM prefers low values of  $g_j$  (i.e.  $g_j^* = \underline{g}_j$  or  $g_{j*} = \bar{g}_j$ ).

The UTA (UTilité Additives) methods are based on the true criterion. It means that, in order to establish the relationship  $a_t > a_s$ , a system of linear inequalities must be considered in the following manner:

$$\begin{cases} g_j(a_t) > g_j(a_s) \quad \forall g_j : g_j^* = \bar{g}_j \\ g_j(a_t) < g_j(a_s) \quad \forall g_j : g_j^* = \underline{g}_j \end{cases} \quad (1)$$

And, in order to establish the relationship  $a_k \sim a_n$ , the equality  $g_j(a_k) = g_j(a_n)$  must hold for all criteria.

The UTA methods solves the multiple criteria ranking problem, which consists in ordering the alternatives from the best one to the worst one, by comparing their global utility values [4]. In the UTA methods, the idea is to described the global utility function as the sum of the marginal utility functions and a residual value, which defines an estimation error. For the alternative  $a_k$  this error measures the distance between the theoretical utility value, obtained by the utility model and the empirical utility value, provided by DM, of  $a_k$ , as follows:

$$\varepsilon(a_k) = \left| U(a_k) - \sum_{j=1}^m u_j[g_j(a_k)] \right| \quad (2)$$

The DM expresses his/her preferences in terms of utility values on the *reference alternatives*, belonging to the *reference set*  $A_R$ . So, taking into account the structure of the true criterion, for each pair of consecutive *reference alternatives*, either the strict preference binary relationship or indifference binary relationship must hold. In order to find them, utility values have to be assigned by the DM, as follows:

- if  $U^*(a'_k) = U^*(a'_{k+1})$ , then  $a'_k \sim a'_{k+1}$
- if  $U^*(a'_k) > U^*(a'_{k+1})$ , then  $a'_k > a'_{k+1}$

With this representation of the DM's preferences, the chain of the binary relationships has the form of a weak order. In the case of a strict preference relationships ( $>$ ), the DM is asked to assign different utility values. In case of uncertainty of the DM, it is sufficient that he/she fixes a value  $\delta$ , defined as follows:  $\delta = U(a'_k) - U(a'_{k+1})$ ,  $\forall (a'_k, a'_{k+1}) \in A_R : a'_k > a'_{k+1}$ .

The marginal utility function  $u(g_j)$  must be monotonic, in order to maintain the property of monotonicity of the true criterion [4]. In particular,  $u(g_j)$  is approximated with piecewise linear function, of which the segments connect the breakpoints  $(g_j^i, u_j(g_j^i))$ . The abscissas of these points are found, as follows:

$$g_j^i = \underline{g}_j + \frac{i-1}{\beta_j-1} (\bar{g}_j - \underline{g}_j), \forall i = 1, \dots, \beta_j \quad (3)$$

where  $\beta_j$  is the number—fixed by DM—of the breakpoints of the marginal utility function  $u(g_j)$ . The values  $g_j^i$  in (3) divides the interval  $[\underline{g}_j, \bar{g}_j]$  of the criterion  $g_j$  into  $\beta_j - 1$  subintervals, having equal size. Clearly, when a criterion  $g_j$  is of gain type,  $g_j^1 = \underline{g}_j$  and  $g_j^{\beta_j} = \bar{g}_j$ .

The ordinates  $u_j(g_j^i)$  are the marginal utility values to be estimated by means of the Linear Programming (LP). As a consequence, the marginal utility values of each performance  $g_j(a_k)$  must be computed as linear function of  $u_j(g_j^i)$  by using the linear interpolation, according to the following formula:

$$u_j[g_j(a_k)] = u_j(g_j^i) + \frac{g_j(a_k) - g_j^i}{g_j^{i+1} - g_j^i} [u_j(g_j^{i+1}) - u_j(g_j^i)] \quad (4)$$

where  $[g_j^i, g_j^{i+1}]$  is the interval, containing the value  $g_j(a_k)$  on the criterion  $g_j$ .

Let  $\Delta(a_k, a_{k+1})$  be the difference between the two theoretical global utility values  $U(a_k)$  and  $U(a_{k+1})$ . For all pairs  $(a'_k, a'_{k+1}) \in A_R : a'_k > a'_{k+1}$ , their corresponding empirical utility values, assigned by the DM, must be compared with their theoretical utility values, in the following manner:

$$\Delta(a'_k, a'_{k+1}) = U(a'_k) - U(a'_{k+1}) \geq \delta \quad (5)$$

where

$$U(a'_k) = \sum_{j=1}^m \left\{ u_j(g_j^i) + \frac{g_j(a'_k) - g_j^i}{g_j^{i+1} - g_j^i} [u_j(g_j^{i+1}) - u_j(g_j^i)] \right\} + \varepsilon(a'_k).$$

Given a *reference set*  $A_R$ , composed of at least two alternatives, the marginal utility values  $u_j(g_j^i)$  and the estimation errors  $\varepsilon(a'_k)$  are estimated by solving the following LP problem:

$$\begin{aligned} \min z &= \sum_{a'_k \in A_R} \varepsilon(a'_k) \quad \text{subject to} \\ \Delta(a'_k, a'_{k+1}) &\geq \delta & \forall (a'_k, a'_{k+1}) \in A_R : a'_k > a'_{k+1} & (6.1) \\ \Delta(a'_k, a_{k+1}) &= 0 & \forall (a'_k, a_{k+1}) \in A_R : a'_k \sim a_{k+1} & (6.2) \\ \sum_{j=1}^m u_j(g_j^*) &= 1 & & (6.3) \\ u_j(g_{j^*}) &= 0 & \forall j = 1, \dots, m & (6.4) \\ u_j(g_j^{i+1}) - u_j(g_j^i) &\geq 0 & \forall j = 1, \dots, m, \forall i = 1, \dots, \beta_j - 1 & (6.5) \\ u_j(g_j^i) &\geq 0 & \forall j = 1, \dots, m, \forall i = 1, \dots, \beta_j & (6.6) \\ \varepsilon(a'_k) &\geq 0 & \forall a'_k \in A_R & (6.7) \end{aligned}$$

The two normalized constraints (6.3) and (6.4) are necessary to find the marginal utility values in the interval  $[0, 1]$ .

The constraint (6.5), regarding to an increasing utility function is necessary to maintain the property of monotonicity of the true criterion [4]; but, the phenomenon  $u_j(g_j^{i+1}) = u_j(g_j^i)$ , when  $g_j^{i+1} > g_j^i$  can occur. It means that the marginal utility functions can be piecewise linear with steps.

The optimal objective function value is reached when  $z^* = 0$ , if all the estimation errors are equal to 0. In this case, multiple optimal solutions can be found; they lead to a perfect representation of the same weak order [4]. While, in the case  $z^* > 0$ , the corresponding optimal solutions are accepted, if they satisfy the Kendall's index [4]. This statistical index measures the concordance between the ranking provided by the DM and those obtained llobal utility function.

In order to overcome the non uniqueness of the optimal solution and the situation  $z^* > 0$ , the suggestion is to solve  $2m$  LP problems, in the phase called



post-optimality analysis [4]. Then, the final solution is computed as the arithmetic mean of the two solutions, obtained by solving the following problems  $LP_j^{min}$  and  $LP_j^{max}$  for  $j = 1, \dots, m$ :

$$\begin{array}{ll} \min u_j(g_j^*) & \text{subject to} \\ z^* - z \geq k(z^*) & (7.1) \\ \text{constraints} & (6.1) - (6.7) \end{array} \quad \begin{array}{ll} \max u_j(g_j^*) & \text{subject to} \\ z - z^* \leq k(z^*) & (7.1) \\ \text{constraints} & (6.1) - (6.7) \end{array}$$

where  $k(z^*)$  is a positive threshold which is a small proportion of the  $z^*$  [4].

An improved version of the UTA method is the UTASTAR method [4]. This method distinguishes the estimation error in the overestimation error  $\varepsilon^+$  and the underestimation error  $\varepsilon^-$ . Moreover, UTASTAR uses the variables  $w_{ji}$ , defined as follows  $w_{ji} = u_j(g_j^{i+1}) - u_j(g_j^i)$ , from the constraint (6.5).

Hence, the LP problem for the UTASTAR method becomes:

$$\begin{array}{ll} \min f = \sum_{a'_k \in A_R} \varepsilon^+(a'_k) + \varepsilon^-(a'_k) & \text{subject to} \\ \Delta(a'_k, a'_{k+1}) \geq \delta & \forall (a'_k, a'_{k+1}) \in A_R : a'_k > a'_{k+1} \quad (8.1) \\ \Delta(a'_k, a'_{k+1}) = 0 & \forall (a'_k, a'_{k+1}) \in A_R : a'_k \sim a'_{k+1} \quad (8.2) \\ \sum_{j=1}^m \sum_{i=1}^{\beta_j-1} w_{ji} = 1 & (8.3) \\ w_{ji} \geq 0 & \forall j = 1, \dots, m, \forall i = 1, \dots, \beta_j - 1 \quad (8.4) \\ \varepsilon^+(a'_k) \geq 0, \varepsilon^-(a'_k) \geq 0 & \forall a'_k \in A_R \quad (8.5) \end{array}$$

The optimal objective function value is reached when  $f^* = 0$ , if all the estimation errors are equal to 0.

In the case of non uniqueness of the optimal solution and when  $f^* > 0$ , the suggestion is to compute the final solution as the arithmetic mean of the  $m$  solutions, obtained by solving the following problems  $LP_j^*$ , for  $j = 1, \dots, m$ :

$$\begin{array}{ll} \max \sum_{i=1}^{\beta_j-1} w_{ji} & \text{subject to} \\ \sum_{a'_k \in A_R} [\varepsilon^+(a'_k) + \varepsilon^-(a'_k)] - f^* \leq \xi & (9.1) \\ \text{constraints} & (8.1) - (8.5) \end{array}$$

where  $\xi$  is a very small positive value, fixed arbitrarily by DM [4].

The number of all the LP problems, including those in the post-optimality analysis, is equal to  $2m + 1$  for the UTA method and  $m + 1$  for the UTASTAR method. It is evident that the UTASTAR method is preferable to the UTA method. But, both the methods with their post-optimality analysis are considered in the next application, in order to compare the final rankings.

In the next section, the analysis of an application in the web context is presented.

**Table 1** Performances of the eight learners on the six indicators

Learner	$I_1$	$I_2$	$I_3$	$I_4$	$I_5$	$I_6$
L	0.83	0.53	0.89	0.01	0.53	0.01650
NB	0.80	0.46	0.87	0.00	0.46	0.99490
RF	0.83	0.53	0.90	0.01	0.55	0.00006
BA	0.82	0.44	0.90	0.03	0.48	0.11520
BOO	0.81	0.50	0.88	0.00	0.50	0.56530
NN	0.82	0.52	0.89	0.01	0.52	0.10180
SVM	0.83	0.59	0.89	0.01	0.57	0.00018
SLAD	0.87	0.64	0.92	0.01	0.65	$<2e^{-16}$

### 3 An Application of the UTA and UTASTAR Methods

In this section, an application of the UTA and UTASTAR methods in the web context is presented.

In this application, the input data are the result of an experiment which was carried out by the Italian National Institute of Statistics and by other member states in the EU [5]. The aim of this experiment was to predict the values Yes/No of the variable “On line ordering or reservation or booking (shopping cart)”, inserted in the “Survey on ICT usage and e-Commerce in Enterprises”, by using some machine learning techniques, as the Logistic (L) model, the Naïve Bayes Network (NB), the Random Forest (RF), the Bootstrap criterion or Bagging (BA), the Boosting (BOO), the Neural Network (NN), the Support Vector Machine (SVM) and the Statistical and Logical Analysis of Data (SLAD) learners.

The learner SLAD was proposed in [6] as an improved method of the classical Logical Analysis of Data.

The performances of all the learners were evaluated by the following indicators:

$I_1$  is the Accuracy, which is the rate of the correct predictions on the total number of the units;

$I_2$  is the Sensitivity, which is the rate of the *true positives* on the total number of the positives;

$I_3$  is the Specificity, which is the rate of the correct *true negative* on the total number of the negatives;

$I_4$  is the difference between the proportion of the observed positives on the total number of the units and the proportion of the predicted positives on the total number of the units;

$I_5$  is the harmonic mean, which is the harmonic mean of the recall and the precision indicators;

$I_6$  is the p-value related to the test Accuracy > Non Informative Rate [5].

The matrix of the input data describes the performances of the eight learners on the six indicators, as reported in the Table 1.

**Table 2** Global utility values for all the learners, computed by UTA and UTASTAR

Learner	UTA	Learner	UTASTAR
SLAD	0.88883	SLAD	0.88883
SVM	0.61015	RF	0.83792
RF	0.57208	SVM	0.83001
L	0.53891	L	0.76924
NN	0.45321	NN	0.68619
BA	0.40087	BOO	0.58802
BOO	0.16670	BA	0.46032
NB	0.16670	NB	0.20000

The DM must fix the number of the breakpoints for each criterion, must choose the *reference alternatives* and must fix the value  $\delta$  (or the utility values of each *reference alternative*). For each criterion, three breakpoints were fixed, of which the abscissas are  $g_j^1, g_j^2, g_j^3$ , where  $g_j^2$  is the middle value of  $g_j^1 = \underline{g}_j$  and  $g_j^3 = \bar{g}_j$ . The value  $\delta$  was fixed equal to 0.01; but the same results were found by fixing it equal to 0.001 and 0.01.

In this application, no preference information was provided by the DM.

In absence of preference information, all the possible weak orders must be individuated in an objective way, following the structure of the true criterion. Let  $\Omega_R = \{A_{R_1}, A_{R_2}, \dots, A_{R_\omega}\}$  be the set of all the possible *reference sets*. With multiple *reference sets*, the total number of all the LP problems, including those in the post-optimality analysis, is equal to  $|\Omega_R| \cdot (2m + 1)$  for the UTA method and to  $|\Omega_R| \cdot (m + 1)$  for the UTASTAR method.

Considering that  $I_1, I_2, I_3, I_5$  are gain criteria and  $I_1, I_2$  are cost criteria, for each pair of alternatives of the input data, there is no indifference binary relationship, while there is only one strict preference binary relationship  $SLAD > BA$ , according to the system (1). So,  $\Omega_R = \{A_R\}$  where  $A_R = \{SLAD, BA\}$ . For the unique *reference set*, thirteen and seven LP problems must be solved for the UTA method and for the UTASTAR method, respectively. From their results, the global utility values are computed and reported in the Table 2. In this table, the learners are ranked, from the highest utility value to the lowest utility value.

The first comment is that the learner Naïve Bayes is always ranked in the last position, with the lowest utility values, while the SLAD learner is always ranked in the first position, with the highest utility values.

It is interesting to note that it is possible to individuate three clusters: one in the head ( $C_1$ ), one in the body ( $C_2$ ) and one in the tail ( $C_3$ ) of the final rankings. The meaning of each cluster seems to be evident. The cluster  $C_1$  contains the best learners, the cluster  $C_3$  contains the worst learners and the cluster  $C_2$  the remaining ones.

In this application, comparing the two final rankings, the SLAD, the SVM and the Random Forest learners can be considered the best learners and the Boosting,

Bagging and Naïve Bayes can be considered the worst learners. As said in the introduction, the last three learners can be deleted from the DM's decisional process.

But, the DM can decide to individuate only two cluster: one in the tail, containing the alternatives to be disregarded and one in the head, containing the remaining ones to be considered for further analysis.

It can occur that  $\Omega_R$  is an empty set. In this case, in order to find a weak order in an objective way, the system (1) must be analyzed, for all the pairs of alternatives. The criterion  $g_s$  with the highest frequency of not satisfied inequalities cannot be considered for searching the weak order. The suggestion is to leave out the criterion  $g_s$  from the search of the weak order. Anyway, this criterion  $g_s$  can participate to the construction of the global utility values.

## 4 Conclusions

The aim of this paper is to propose an efficient methodology for solving, in an easy and quick way, the problem of model selection, for all types of models, whose performances can be measured by indicators and indices.

The problem of model selection can be viewed as a ranking problem in the MCDA context. Among the methods solving the ranking problem [7], the UTA methods are well suited, because they construct a unique criterion, which synthesizes the indicators or indices, that can be conflicting. The unique criterion, describing the global utility values of each models under consideration, is constructed by means of linear programming problems. The use of linear programming guarantees global optimal solutions; moreover, in the post-optimality analysis, the unique global optimal solution is guaranteed.

In this paper, also a proposal of a methodology when the DM is not able to provide any information or judgments on at least two alternatives, is suggested. Such a methodology consists in finding all the possible weak orders in an objective way, according to the system (1). Then, for the corresponding each *reference set*, all the LP problems, including those in the post-optimality analysis, must be solved. This process is necessary in order to have a global knowledge of the information provided by the input data only.

In the future works, the other variants of the UTA methods family [4, 8] will be used, in order to improve the methodology suggested in this short paper.

## References

1. Kadane, J.B. Lazar, N.A.: Methods and criteria for model selection. *J. Am. Stat. Assoc.* **99**(465) (2004)
2. Zucchini, W.: An introduction to model selection. *J. Math. Psychol.* **44**, 41–61 (2000)
3. Mousseau, V., Slowinski, R., Zielniewicz, P.: ELECTRE TRI 2.0, a methodological guide and users manual. Document du LAMSADE no111, Universit Paris-Dauphine (1969)

4. Siskos, Y., Grigoroudis, E., Matsatsinis, N.: Multiple criteria decision analysis: state of the art survey. In: Figueira, J., Greco, S., Ehrgott, M.(eds.), pp. 297–344. Springer, Boston, Dordrecht, London (2005)
5. Barcaroli, G., Bianchi, G., Bruni, R. Nurra, A., Salomone, S., Scarnó, M.: Machine learning and statistical inference: the case of Istat survey on ICT. In: Proceedings of the 48th Scientific Meeting of the Italian Statistical Society, Salerno, June 2016
6. Bruni, R., Bianchi, G.: Effective Classification using Binarization and Statistical Analysis. *IEEE Trans. Knowl. Data Eng.* **9**, 2349–2361 (2015)
7. Guitouni, A., Martel, J.M.: A multidimensional evaluation of the effectiveness of business incubator: an application of the PROMETHEE outranking method. *Eur. J. Oper. Res.* **109**, 501–521 (1998)
8. Greco, S., Mousseau, V., Slowinski, R.: Ordinal regression revisited: multiple criteria ranking using a set of additive value functions. *Eur. J. Oper. Res.* **191**, 415–435 (2008)

# The Importance to Manage Data Protection in the Right Way: Problems and Solutions

Hassan Mokalled, Daniele Debertol, Ermete Meda and Concetta Pragliola

**Abstract** Data has become the most important asset for the companies, and data protection against loss is fundamental for their success. Most of the companies are connected to internet for business reasons and this is potentially risky. Cyber-attacks, hacks and security breaches are no longer an exception Arora et al. (Empir Anal Inf Syst Front 8(5), 350–362, [1]). They can range from no or limited impact to Distributed Denial of Services (DDoS), stealing/manipulation of data, or even taking over control of systems and harm the physical world Andrew et al. (Decision Support Approaches for Cyber Security Investment, [2]). Some companies work on critical projects that contain documentation to be protected and not publicly disclosed. Data leakage or loss could lead to hazardous situations, so data confidentiality, integrity and protection should be conserved. To reach this goal, it is better to adopt an efficient data protection management, i.e. having effective processes and methodologies in place to enable prevention, detection and reaction to any threat that could occur. Companies should give importance to actions, plans, polices, and address the organizational aspect, and be aware and prepared to manage crisis situations, using the best technological solution for each stage of the cybersecurity management. In this paper, we present solutions and key steps to

---

H. Mokalled  
DITEN, University of Genova, Genoa, Italy  
e-mail: Hassan.Mokalled.ext@ansaldo-sts.com

H. Mokalled · D. Debertol · E. Meda  
Ansaldo STS, Cyber Security Assurance & Control, Genoa, Italy  
e-mail: Daniele.Debertol@ansaldo-sts.com

E. Meda  
e-mail: Ermete.Meda@ansaldo-sts.com

H. Mokalled  
EDST- MECRL Lab, Lebanese University, Beirut, Lebanon

C. Pragliola (✉)  
Ansaldo STS, Cyber Security Assurance & Control, Naples, Italy  
e-mail: Concetta.Pragliola@ansaldo-sts.com

manage data protection inside Ansaldo STS Company from organizational and technological sides, by using an Information Security Management System that implements the cybersecurity strategy of the company through three phases (prevention, detection and reaction, and checks for compliance and improvement) and by adopting a defense-in-depth approach and maturity models to deploy control in a prioritized and effective way.

**Keywords** Data protection • Cybersecurity • ISMS

## 1 Introduction

The use of computers, storage, networking and other devices to create, store and exchange all forms of data has increased significantly in the last years. Interconnectivity and data generated by devices has resulted in ‘an unprecedented improvement in the quality of life’ [3]. At the same time, the vast amount of data available about activities is giving rise to cybersecurity and privacy challenges. Data can be said to be the most valuable asset that companies strive to protect. Data, such as technical and non-technical documentation, financial and health records, and intellectual property may be worth millions of euros in the hands of hackers and data thieves. If organizations and companies do not address data security issues, critical threats to information privacy may develop. Businesses and other organizations thus must take action to secure the sensitive data they control [4]. With the diffusion of the internet and new storage media, data may be compromised on a larger scale and at a faster pace. With the sharing of data on networks, a threat to data security is becoming a major concern. Protection of information is necessary to establish and maintain trust between an institution and its stake holders. Protecting data is not just a technology issue anymore [1]. Entire management systems inside companies now are giving enormous attention to organizational aspect. Policies, proved objectives, audits, training and awareness activities, compliance with legal and regulatory requirements for security and privacy have become important factors to be addressed in information security. One of the main requirements toward all of this stands the assessment of risk and its evaluation [5]. Consequently, companies must realize the necessity of paying attention to the organizational aspects of data protection. And so managing data protection can be better treated addressing two points of view: the organizational and the technological ones. Ansaldo STS is a leading company operating in the sector of high technology for railway and urban transport. The Company has the experience and resources to supply innovative transport systems for freight yards, regional and freight lines, underground and tramway lines, and standard and High-Speed railway lines. With an international geographical organization, The Company operates worldwide as lead contractor, system integrator and supplier “turnkey” of the most important projects of mass transportation in metro and urban railways. Ansaldo STS has a great experience in the design, implementation and management of systems and services for signalling

and supervision of railway and urban traffic. For the described goal, Ansaldo STS adopts an information security strategy implemented by an Information Security Management System (ISMS) which describes the organizational aspects of data protection inside the company, adopting the governance, risk and compliance approach. From a technological point of view, Ansaldo STS adopts a defense-in-depth approach and maturity models to deploy the security controls in a prioritized and effective way in accordance to the organization's overall strategies and policies. This paper is divided as follows: the next section describes the aspects of data protection, the third section presents the strategy adopted by Ansaldo STS to protect its data, the fourth section is about the experience of Ansaldo STS and its ISMS, and section five is the conclusions.

## **2 Data Protection Aspects**

Information and communication technology (ICT) has made remarkable impact on the society, especially on companies and organizations. The use of computers, databases, servers, and other technologies has made an evolution on the way of storing, processing, and transferring data. However, companies access and share their data on internet or intranet, thus there is a critical need to protect this data from destructive forces and from the unwanted actions of unauthorized users. To design a solution that truly protects the data, we must understand the security requirements relevant to our site, and the scope of current threats to our data [6, 7]. To correctly manage data protection, it is important to take in account some aspects in the procedure of data protection, the main aspects are:

- i. Data classification levels.
- ii. Threats and Vulnerabilities.
- iii. Data security requirements.

### ***2.1 Data Classification Levels***

Data is one of the strategical components of the corporate assets essential to a company. For this reason, it should be protected within a company in accordance with its own value and its significance to the company's business by implementing a classification process. Data classification is also useful to identify who should have access to the technical data used to run the business versus those who are permitted to access test data and programs under development. Data classification must take into account legal/regulatory/internal requirements for maintaining confidentiality, integrity and availability. Data classification should define:



- a. The owner of the information asset
- b. Who has access rights (need to know)
- c. The level of access to be granted
- d. Who is responsible for determining the access rights and access levels
- e. Which approvals are needed for access
- f. The extent and depth of security controls.

However, data shall be classified by means of a method entailing an established structure of criticality and protection levels, which shall be determined in accordance with the potential impact on the company (e.g. the economic value, the damage to the company's reputation, the legal constraints and the strategical significance). An example of the classification levels defined in a decreasing order of criticality can be as follows:

- **PRIVILEGED:** Data that has not to come to the public domain, because might, if reaching the public domain, affect the prices of such financial instruments to a significant extent.
- **CONFIDENTIAL:** Data concerning the company and/or its own subsidiaries, which may, when disclosed freely, cause an economical damage or affects the Company's reputation.
- **RESTRICTED:** Data that can be freely accessed by the personnel working at the Company. This is the default classification level that shall be assigned every time a new piece of information is created.
- **PUBLIC:** Data that can be freely disclosed outside the Company, since its disclosure shall cause no damage to the Company itself.

## 2.2 *Threats and Vulnerabilities*

The two main kinds of security threats that affect a company are internal and external threats. Internal threats occur from within the organizations. This is probably one of the most dangerous situations because for instance co-workers may know passwords to access systems and are aware of how the systems are set up. Computers that are left unattended can be easily accessed by workers. And external threats are attacks done by hackers [6].

- *Internal Threats:* Previous research on cybersecurity has focused on protecting valuable resources from attacks by out-siders. However, statistics [8, 9] show that a large amount of security and privacy breaches are due to insiders. Protection from insider threats is challenging because insiders may have access to many sensitive resources and high-privileged system accounts. Similar style of exploitation is reported in [10, 11].
- *External threats:* External threats are those done by individuals from outside a company or organization, who seeks to break defenses and exploit vulnerabilities. Spying or eavesdropping, DoS, Spoofing, Phishing, viruses, etc., are all

examples of external threats or cyber-attacks. However, one of the emerging threats and the latest criminal invention is the Ransomware which is a special type of virus that does not destroy any data but simply encrypts all the data it finds on a PC with an encryption key that only the criminal has, and asks for money to give the key.

- *Vulnerabilities*: Weakness in the organization or company cyber-assets that a malicious attacker could use to cause damage. Vulnerabilities could exist in system, installed software, and network.

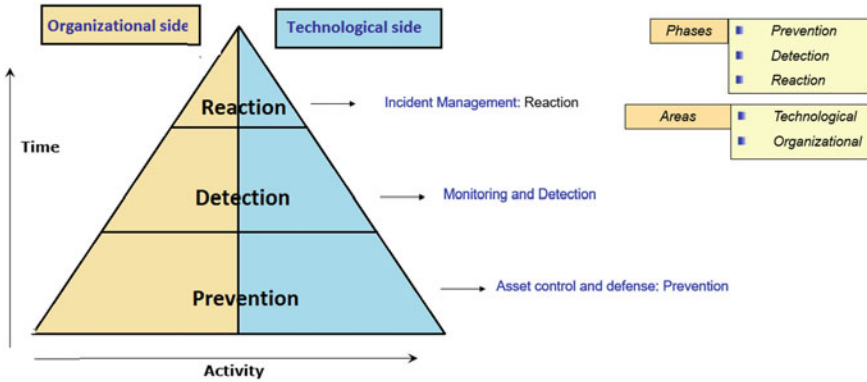
### 2.3 Data Security Requirements

Data security concerns the use of a broad range of information security controls to protect the whole system (potentially including the data, the applications or stored functions, the database systems, the database servers and the associated network links) against compromises of their confidentiality, integrity and availability [6]. Data protection must address these main security requirements:

- *Confidentiality*: This means that data must not be exposed to unauthorized individuals. And access must be restricted to those authorized to view the data. Confidentiality has several different aspects: privacy of communications, securing storage of data, authentication of users, and access control.
- *Integrity*: Data integrity means that data should be protected from corruption while it is stored in the database or transmitted over the network. Integrity has different aspects: only authorized users can change data, protecting the network and data against viruses designed to corrupt or delete.
- *Availability*: Data must be available to authorized users, without delay. Denial-of-service attacks are attempts to block authorized users' ability to access and use the system when needed [6].

## 3 A Data Protection Management Approach

Data protection aims at protecting the Confidentiality, Integrity and Availability (CIA) of company data and information whether it is processed, transmitted, stored on and/or in transit through networks and systems. Data protection involves the protection of all the cyber-space used in the company to store, process and transfer this data against unauthorized use, disclosure, transfer, modifications or destruction, whether accidental or intentional, or the loss of availability of these assets or business processes to authorized users. Data protection is ensured by an information security (cybersecurity) strategy used to secure all assets involved in the storage,



**Fig. 1** Data protection strategy process

processing, transmission of data such as databases, computers, servers, network devices, etc. The need for cybersecurity is becoming increasingly important due to our dependence on Information and Communication Technology (ICT) to store, process and transmit data. Companies do not want to be associated with cybersecurity hacks or viewed as having not taken appropriate security measures [2]. On the other hand, different types of threats and vulnerabilities that threatens company data varies between internal and external ones, and this requires different types of countermeasures starting by setting plans, policies, complying to laws and standards, training internal staff, and so on, and also setting appropriate countermeasures. For this reason, data protection requires a strategy that covers both organizational and technological security aspects inside a company, applied on the cybersecurity phases of prevention, detection, and reaction (Fig. 1).

From the organizational point of view, it is essential to set policies, actions, plans, responsibilities and ensure audits. On the technological side, the goal is to specify and implement the selected controls in the policies against threat scenarios. Both aspects should cover all phases of cybersecurity:

- i. *Prevention phase*: proactive phase for the defense of company assets.
- ii. *Detection phase*: monitoring of company assets.
- iii. *Reaction phase*: incident management.

In this section, we aim to describe the data management approach designed by Ansaldo STS aiming to protect the company's data both from the organizational and the technological side, and which was inspired by the ISO 27001 requirements. This approach is implemented by an ISMS from an organizational aspect, and follows the Governance, Risk, and Compliance (GRC) framework. And from the technological aspect, a defense in depth approach is adopted to deploy the controls selected by type and in a prioritized and effective way.

### ***3.1 Using an Information Security Management System (ISMS) Based on the GRC Framework***

An Information Security Management System (ISMS) consists of the policies, procedures, guidelines, and associated resources and activities, collectively managed by an organization, in the pursuit of protecting its information assets. An ISMS is a systematic approach for establishing, implementing, operating, monitoring, reviewing, maintaining and improving an organization's information security to achieve business objectives. The goal of ISMS is to minimize risk and ensure continuity by pro-actively limiting the impact of a security breach [12]. The ISMS shall be balanced and integrated into the daily actions of employees; in addition, it shall be balanced among business goals, productivity and ensuring adequate data protection levels of the company and it shall ensure the privacy of employees. Business and IT staff, relevant for information security activities, shall be trained in order to ensure the application of the defined ISMS, and awareness initiatives shall be deployed to all employees. For this purpose, Ansaldo STS implements an ISMS and related documents are created, developed and published as means to implement data protection strategy, in accordance with the company business requirements, strategies, and relevant laws, regulations, and contractual agreements. This ISMS is based on a governance, risk and compliance framework.

- a. **Governance:** Governance activities involve setting objectives to achieve and defining a way to achieve them while maintaining transparency with internal and external stakeholders. Governance tools, such as system controls and policies, are implemented in order to ensure that processes are followed in a proper manner. The Governance includes all activities necessary to define and implement a framework aimed at ensuring a proper data protection management. The main activities should define: roles and responsibilities, processes, policies and procedures (including supporting and monitoring tools), audit plans. Main activities reported above must be executed according to the Segregation of Duties (SoD) principle. SoD aims at avoiding situations where a single person could execute or control several phases of the same process, or different processes identified as incompatible. The aim is to mitigate potential exposures to human mistakes or fraud events. Correct implementation of data protection strategy is verified through periodical audits performed by IS departments and by third parties.
- b. **Risk:** Risk Management activities involve risk identification, assessment and mitigation plan definition. All risk management activities shall be performed on an ongoing basis in order to ensure that new risks are identified and previous identified risk are mitigated. Risk management acts as an internal control system that has to grow together with the business growth. The process of assessment, management and monitoring of risks through establishing and maintaining an appropriate risk framework is performed by the IS Manager. The assessment allows to identify the action plan to put forth mitigating the identified risks and protect the Confidentiality, Integrity and Availability of assets.

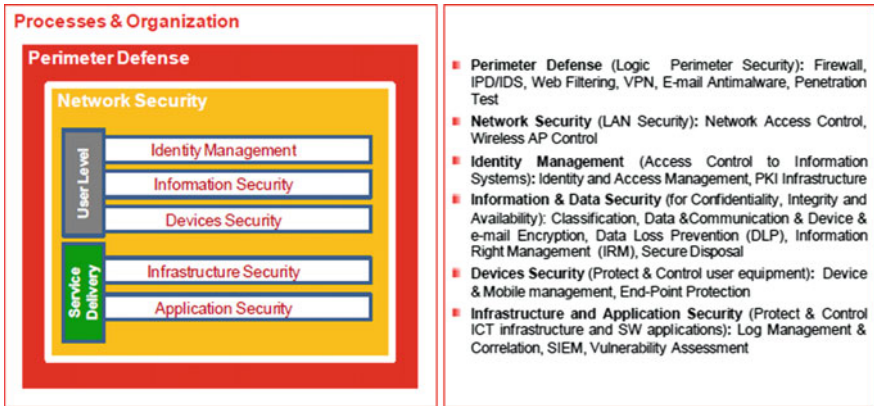
- c. **Compliance (Compliance to international standards and norms):** Compliance activities involve regulatory analysis in order to ensure the compliance with global and local applicable laws. Applicable laws are identified on the basis of the regulatory framework applicable to industry and country in which the company operates. Compliance can be oriented to internal policies and rules or to external laws and regulations, but, in any case, it represents a fundamental step in order to maintain the organization control inside its specific regulatory environment. In this context, compliance shall:
- be maintained with all applicable national and international privacy legislation, and with international information security standards such as ISO/IEC 27001, GDPR (General Data Protection Manager) or other equivalent best practice/regulation required by the business;
  - ensure that all employees/third parties follow all security requirements.
  - ensure that all employees and collaborators as well as third parties with access to information systems are aware of their responsibility to report any security incident as quickly as possible.

### ***3.2 Technological Point of View-Defense in Depth Approach***

Data protection from a technological side is about executing the ISMS plans and operations, by selecting the right counter measures, specifying their types, prioritizing them by maturity levels, day to day operation, etc. To fully protect the data during its lifetime, each component of the information system must have its own protection mechanisms. The building up, layering on and overlapping of security measures is called defense in depth.

#### **3.2.1 Defense in Depth Approach**

Defence in Depth (DiD) is an efficient operational approach that enables to manage (by a risk-oriented approach) people, processes and technology. In IT environments, DiD is intended to increase the costs of an attack against the organization, by detecting attacks, allowing time to respond to such attacks, and providing layers of defense so that even successful attacks will not fully compromise an organization. A DiD strategy is necessary because of new security threats and the importance of IT security monitoring of assets. Main variables that have increased the importance of DiD strategy definition are for example: the increased value of data, globalization, mobile working, virtualization, and decentralization of services. In this context, the company shall recognize the need to provide coordinated and multi-layered security architectures to mitigate security risks. Implementing DiD requires an understanding of enterprise strategy, applicable internal and external threats,



**Fig. 2** Defense in depth: Layering and setting technologies

information asset classification, and technology supporting controls. The Defense in Depth strategy shall define all layers and technologies which, given company environment and security requirements, are necessary for all the different parts of the organization, as depicted in the figure below. Referring to Best Practice and Guidelines, Ansaldo STS adopted the DiD by using five levels to describe security actions based on the plans and policies of the ISMS (Fig. 2).

**3.2.2 Maturity Levels**

Referring to Maturity Model, Ansaldo STS intended their policies to be applicable completely although at a different pace, so they divided them according their complexity (to verify where controls would go deeper and where has been less deep). The maturity levels are used to set the level of security and control of a specific configuration or solution. The Maturity Model adopted by the company includes the following maturity levels:

**L1: INITIAL**

- Minimum set of acceptable security measures and control activities are designed and in place.
- Security measures and control activities have been documented and communicated to stakeholders and interested parties.

**L2: IMPROVED**

- Standardized controls with periodic testing for effective design and operation with reporting to management are in place.

- Improved or selected security measures are in place to harden specific controls or business areas.
- Automation and tools may be used in a limited way to support control activities and security measures effectiveness.

### **L3: OPTIMIZED**

- An integrated control framework with real-time monitoring by management for continuous improvement (enterprise-wide risk management) of the security measures is in place.
- Automation and tools are widely used to support control activities and allow the organization to make rapid changes to the security measure in place.
- High level of security measures are available, addressing the trade-off between residual risk and costs.

## **4 The ISMS of Ansaldo STS**

In this section we describe the ISMS designed by Ansaldo STS used to manage data protection inside our company. The ISMS implements the whole information security process used to protect the data within the company. Both information security and information technology departments are involved, and they have the responsibility and accountability of executing the sub process.

### ***4.1 ISMS Process***

The activities of an information security process implemented by the ISMS can be divided into four vertical domains with different responsibilities and accountabilities represented by the company area in Fig. 3:

- ***Governance and Risk***: defining strategy, policy (security levels), requirements (constraints), procedures, and conformity depending on internal or external requirements, laws and international standards. Then evaluate the Risk and identify the countermeasures to be taken to obtain an acceptable level of risk.
- ***Design***: defining information security architectures and technology solutions based on the countermeasures to be adopted and the approved budget in accordance with the defined strategy.

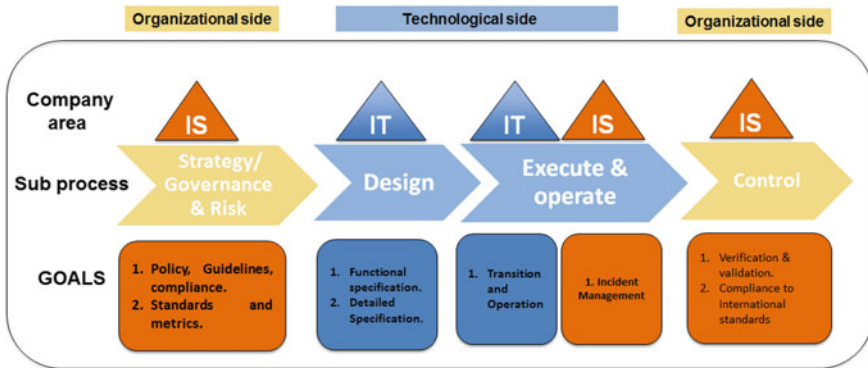


Fig. 3 The IS process implemented by the ISMS for data protection management

- **Operation (Operate and Execute):** putting the activities defined into operations, including the new transitions, the change of existing ones, day-by-day operations and maintenance of the equipment.
- **Control:** assessing the adherence of current levels and security configurations to the policy, requirements and compliance set out in governance phase.

## 4.2 Segregation of Duties

Ansaldo ISMS is developed in accordance to the “segregation of duties” principle as stated by the A6.1.2 control of Annex A in ISO 27001. The A6.1.2 control of the ISO 27001 states that conflicting tasks and areas of responsibility must be separated to reduce the chances of misuse, unauthorized or unintentional modification of the assets of the organization [13]. Each phase has a responsible department within the company (Fig. 3), with the Information Security Department (IS) and the Information Technology Department (IT) being the involved departments.

- **Governance:** IS department has responsibility and accountability for this phase.
- **Design:** The IT Dept. has responsibility and accountability for this phase.
- **Operation:** The IT Department has the responsibility and the accountability for this phase. However, under this phase, the IS Dept. retains responsibility and accountability for the Incident Management task. In case an incident occurs, the IS department tries to understand the incident, find a solution and define the remediation, and finally gives the IT department the procedure to apply.
- **Control:** The IS department has responsibility and accountability for this phase.



### 4.3 Documentation of the ISMS

At the end, the documentation of the ISMS is carried out. There are five main categories of documents to describe the ISMS of Ansaldo STS Company (Fig. 4):

- *Process Description (PRD)*: It is a high level describing the whole ISMS with—and includes all the responsible, accountable, consulted and informed departments involved in the ISMS.
- *Manual (MNL)*: This type of document is a high-level directive. It describes the strategy, governance, and rules.
- *Procedures (PRC)*: These documents indicate who does what. They describe the responsible individuals and their duties.
- *Instructions (INS)*: These are the instructional documents. They give a brief description for the way of operation or installation indication how it is done.
- *Module—Template—Checklist (FOR)*: These are forms or check lists that must be filled for the purpose of requesting a service from the IT department.

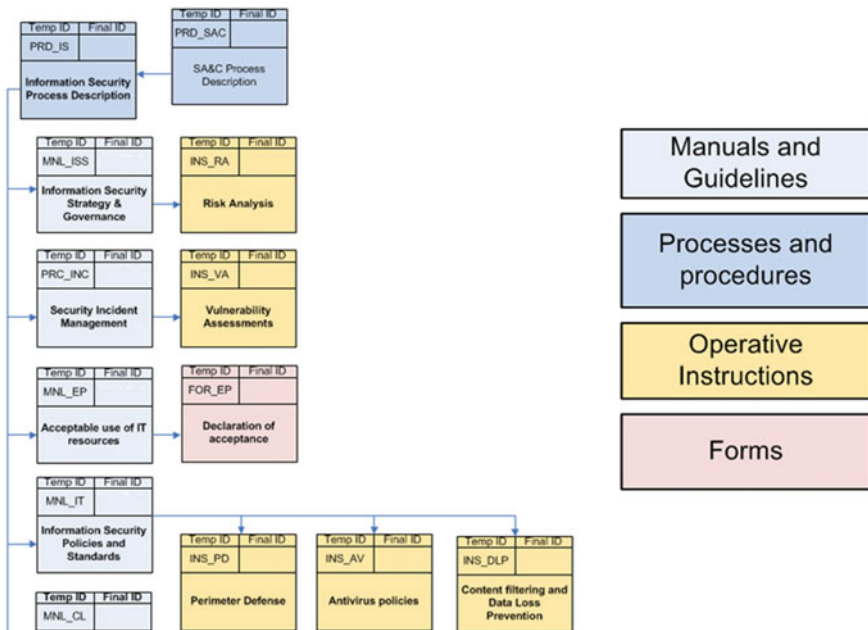


Fig. 4 Documentation of the ISMS

## 5 Conclusion

Data must be treated as the core asset of any organization or company, which should be protected from all kinds of threats. However, to best protect the data, it is better to adopt a good management approach, which minimizes errors, saves time, increases awareness and prepares the company to incidents. In information security, taking due care of strategies, setting policies, plans, preparing and training staff, and complying to internal or external laws and standards should be given the same importance as operating technical tools. The focus should be on both organizational and technical sides. Ansaldo STS company gives a great significance to the organizational side of data protection, which is shown by its ISMS which was created in accordance with the international standards and frameworks. Anyway, Ansaldo STS realizes the necessity to verify that the ISMS is working effectively in the sense that all the defined requirements are correctly implemented, and this can be obtained by regular audits to reach continuous improvement. Ansaldo STS also realizes that not all assets are correctly managed, in fact there are some areas not completely covered such as laboratories, plants connections and so on. And so it is important to extend the coverage of the ISMS to these areas and not only to office areas. In the next future, Ansaldo STS is going to comply with the international standard ISO 27001 on some strategic scopes with the aim to obtain a certification which represents a key goal for the company and its business.

## References

1. Arora, A., Nandkumar, A., Telang, R.: Does information security attack frequency increase with vulnerability disclosure? *Empir. Anal. Inf. Syst. Front.* **8**(5), 350–362 (2006)
2. Andrew F., Emmanouil P., Pasquale M., Chris H., Fabrizio, S.: *Decision Support Approaches for Cyber Security Investment* (2016)
3. Elmaghraby, A.S., Losavio, M.M.: Cyber security challenges in smart cities: safety, security and privacy. *J. Adv. Res.* (2014)
4. Bennett, S.C.: *Data Security Breaches: Problems And Solutions* (2008)
5. The Importance of information security nowadays. [https://pcb.com/pdf/articles/27-pcb\\_the-importance-of-information-security-nowadays.pdf](https://pcb.com/pdf/articles/27-pcb_the-importance-of-information-security-nowadays.pdf)
6. Balvir, S., Amarjeet S.: *A Roadmap to Data Security of Automated University Examination System* (2015)
7. Summers, G.: Data and databases. In: Koehne, H. (ed.) *Developing Databases with Access*, pp. 4–5. Nelson Australia Pty Limited (2004)
8. Annual Emerging Cyber Threats Report, Georgia Tech Information Security Center. <http://www.gtisc.gatech.edu/>, last accessed (2013)
9. Internet Security Threats Report. Symantec. <http://www.symantec.com/threatreport/> (2013)
10. The CERT guide to insider threats: how to prevent, detect, and respond to theft of critical information, sabotage, and fraud. [www.cert.org/archive/pdf/insidercross051105.pdf](http://www.cert.org/archive/pdf/insidercross051105.pdf)

11. Hunker, J., Probst, C.W.: Insiders and insider threats—An overview of definitions and mitigation techniques. *J. Wirel. Mob. Netw. Ubiquitous Comput. Dependable Appl.* **2**(1), 4–27 (2011)
12. International standard ISO\_IEC\_27000 (2014)
13. International standard ISO\_27001 (2013)

# The Use of Configurational Analysis in the Evaluation of Real Estate Dynamics

Enrico G. Caldarola, Valerio Di Pinto and Antonio M. Rinaldi

**Abstract** The shape of urban space together with the choices that lead to its configuration have been the base of long and multidisciplinary debates taking into account several and heterogeneous factors. In this context, the goal of decision makers is to create and improve the value of a given area and manufactures. In this paper we propose a quantitative approach based on configurational analysis in the domain of real estate. The use of geographic information systems to integrate and analyze data form different data sources shows similarities among social-economics models and spatial approaches which consider completely different parameters.

## 1 Introduction

One of the most relevant aspect among urban phenomena is the distribution of real estate values. It is not surprising that many researchers believe that all issues affect the city arise from the lack of knowledge, and the consequent unsuitability of shares, about growth mode of land prices. This statement, however, contrasts with the quan-

---

E.G. Caldarola · A.M. Rinaldi (✉)

Department of Electrical Engineering and Information Technology (DIETI),  
University of Naples Federico II, Via Claudio, 21 - 80125 Naples, Italy  
e-mail: antoniomaria.rinaldi@unina.it; amrinald@unina.it

E.G. Caldarola

e-mail: enricogiacinto.caldarola@unina.it

E.G. Caldarola

Institute of Industrial Technologies and Automation,  
National Research Council, Bari, Italy

V. Di Pinto (✉)

Department of Civil, Architectural and Environmental Engineering (DICEA),  
University of Naples Federico II, Via Claudio, 21 - 80125 Naples, Italy  
e-mail: valerio.dipinto@unina.it

A.M. Rinaldi

IKNOS-LAB Intelligent and Knowledge Systems (LUPT), University of Naples  
Federico II, Via Toledo, 402, 80134 Naples, Italy

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_9

tity of literature. Economists are always engaging real estate rent; almost ever about the way land prices be developed. Theories they have been developing for a century largely disagree. The latter, as simpler empirical models about position rent, generally connect every aspect of urban behavior to a rational selection of simple variables such as distance or economical budget [1]. The outcome is a city picked from a single sight neglecting any information about site or real estate peculiarities. On the other hand, every human urban behavior is handled as the result of efficient and fitting assessments (lengths of time, distances, ...). This condition is the natural outcome of the choice to schematize the city in order to reach reproducible results in the frame of an objective environment, but it is less solid than it seems. The perfect rationality of the users is an arbitrary position and it seems unclear why it is related just to a matter of time and cost. Environmental psychology and social sciences suggest us that a clear optimization of simple variables is not a feature of this kind of experience [2]. Most times seem to take priority other elements, despite hard to be listed, attributable to the link between users and their environment. In light of the above, it is possible to think about a new kind of model approaching real estate price distribution starting from the space, and where it plays a dominant role as generative variable of the city as a system. One of the most innovative theory about urban dynamic concerns to show the outcome of the human perception of the space through a non-local property of the city-system based on the concept of universal distance: the configuration. This theory and related techniques have been developed in a more general framework as known as *Space Syntax* [2, 3]. Configurational analysis is based on the hypothesis of the being of a movement rate, called *natural movement*, only dependent on the shape of the urban grid. More precisely, it depends on the connections between the set of lines of the matrix which is possible to reduce the entire system. Natural movement is considered the primary factor in the spawning of urban movement. At the same time it certainly expresses an indicator of the way users understand the city, interiorise it, use it: percept it. All urban phenomena underlie the rule of natural movement [4]. The main aim of the paper is to verify if the configuration of the city is an effective variable in the distribution of real estate values, proving the existence of a strong correlation between the configurational concept of multi-scalar centrality, the configurational indexes describing such centrality, and the values of the real estate market, patterned as a set of polygons covering the entire city. The work take into account residential estate only, so as to reduce the noise due to the presence and distribution of commercial and tertiary activities working as attractors. With this aim, working on one of the largest Italian urban area, the paper compares the dimension of equal price areas and the parameters resulting from configurational analysis both at global and local level.

Results can be effectively used to support decision processes in urban and transportation domain, for example to predict the real estate patchwork after a major territorial transformation.

In Sect. 2 an analysis of existing literature has been presented and discussed; the proposed methodology is described in Sect. 3 and a complete case study on a real city is shown in Sect. 4; experimental results has been presented in Sect. 5 and conclusions and future works are in Sect. 6.

## 2 Literature Review

During the last two centuries many approaches to the study of land and real estate values distribution have been developed. They have charted an ideal course conditioning our thinking still today, thanks to the functional definition of concepts like rarity and centrality. Contributions can be collected in two differently categories: theories about elements affecting land prices formation, and theoretical models detailing classic economists ideas. They have explained the influence of land rent on urban growth and conversely, although under eased hypothesis, omitting any role of the space. Flourishing of theoretical models was the basement to the growth, especially in the 60's of nineteenth century, of organic theories about real estate prices formation and distribution. They also involved the manner of cities and land uses transformation. The work of Wingo [5] had particular relevance to formalize in a solid mathematical way the relationship between length time and land price. Wider than that is the contribution of Alonso [6]. He proposed to explain land prices formation on the base of spatial economy theories. That affected the basic model hypothesis as smooth plain, and the interests about as real estate as rural marked by twice perspective: families and companies. Alonso's model allows to make predictive assessments on the system of property values to vary one of the parameters, such as income, the cost of transportation, and the intended land use. Ultimately, it is the most organic formalization of a way to understand urban phenomena in an extremely simple, fixed and rational mode. It assumes the hypothesis of monocentricity (Central Business District—CBD). Muth [1], saving a distance decay cost (from CBD) structure, or rather considering accessibility without space as a key variable. It makes CBD exogenous to the model [7]. This is one of the main limitations of traditional models about position rent. The issue is the interpretation of the distance from the CBD as a specific individual estate element quality, dependent on its local urban structure. This would confer CBD the characteristics of an endogenous element into the model. Recent studies investigated the effect of global and local accessibility on office and residential rent and values by using Space Syntax to support a hedonic model. The results suggest the model has good predictive power in explaining the variation in the log of the rent, striving spatial integration and choice indexes [7, 8]. Space Syntax spatial analysis is a set of techniques and theories for the interpretation of configuration of spatial layout in general and urban layout, by using the concept of universal distance under the interpretation of the centrality as a social phenomenon. Recent studies proved that Space Syntax could also take into account urban forms by means of ontologies representing the meanings of city elements [4, 9]. The aim of this paper is to examine residential property value variations of the whole city of Naples using Space Syntax analysis and a database of values built on a set of 3402 normalized transactions registered during 2011-second semester. Different from other approaches previous discussed, our innovation is in the use of quantitative analysis to better understand how configurational centrality at different levels (global and local) affect residential property values distribution.

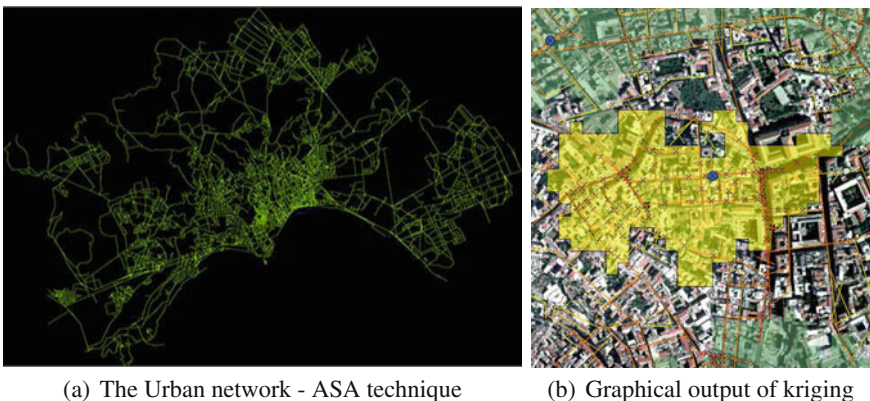
### 3 The Proposed Methodology

The approach of study that is intended to follow to address any starting issues has been chosen according to what was experienced in Space Syntax literature. They were built separately a spatial model, only containing configurational variables, and an information model, which collects territorial and real estate data. Just later, they were joined in a third correlation model. All three models have been constructed or adapted in GIS environment to take advantage of its geo-statistical and informative full potential. Inside the correlation model, two separate regression models have been developed. A global one, studying the influence of the main integrator in the formation of price areas, and a local one, aiming to describe the correlation between configurational indexes and real estate values. The analysis of urban grid configuration was carried out according to techniques and methodologies defined and experimented [3] in the last few years by the Bartlett School. Public spaces of the whole city have been split in the set of the larger and fewest convex spaces. The minimal set of lines that reciprocally correlate them, has been then traced, so as to create a matrix of lines, known as *axial map*. Afterward, the axial map has been processed making use of a specific software provided by the Space Syntax Laboratory of the U.C.L. (DepthMap). Adopting ASA technique [10] configurational indexes have been assigned to every line. Resulting map has then been exported in GIS environment as an ESRI geodatabase mainly hosting feature class of linear typology, storing geometry and configurational informations. The creation of real-estate values Geo-Database has been conversely made by hand, according to available data. Raster maps have been vectorized by drawing their geometry in CAD environment, and then, after the exportation into GIS environment as polygonal feature class, adding non-geographical attributes. In the same information system, territorial data have been added such as orthophotos, the system of public transport, the main thoroughfares, and the contour lines. ArcGIS environment have been used to the regression model also. Inside it, two different models was developed to study how urban grid centrality, at global and local level, relates to real estate values. To better take into account local independence on the grid, the least number of points of polarization of the local integration index were identified by progressively narrowing the metric radius of investigation, from 6.400 to 400 m. The identification of individual points of polarization was obtained by employing the geo-statistical instrument of kriging, identifying, for each variation of the analysis radius, the centroid (midpoint) of each line of the system. In this way, the distribution of the local centrality has been evaluated apart from the simple identification of thresholds for peak.

## 4 The Case Study

The model has been built in step with its application on the case study of Naples, one of the major Italian and Mediterranean city. It is the capital of Campania and, with 960.000 inhabitants living within its administrative boundary, it is the third Italian city after Rome and Milan. The urban area of the city has about 3.5 million inhabitants, about an extra million in the metropolitan area. Naples is thus given as the eighth urban area of the European Union and one of the largest metropolitan areas in the Mediterranean. The administrative area enclosed within the perimeter is approximately 120 km<sup>2</sup>, in view of a very high population density, higher than 8.000 inhabitants/km<sup>2</sup>. Urban morphology is very complex. Along the east-west direction, the city is surrounded by the Mount Vesuvius (a still active volcano) and the Campi Flegrei volcanic complex. Similarly, it is flattened along the north-south from the coast line and a range of hills that divides it from a broad plain at the back, which slopes down towards the sea detaching the urban area into two large flat zones that communicate through tunnel paths. This particular morphology forced the city to grow on itself, presenting today a complex urban maze, heavily layered and historicized. In a context of narrow and intricate spaces, large and spectacular urban arteries of the late nineteenth century unfold their paths, as well recent and still evolving fast mobility and public transport infrastructure.

The complexity of the city is evident from the configurational analysis of its urban grid. It has been extended to the whole urban area, refusing the purely administrative perimeter and identifying the natural limits of the Naples basin. The axial map has 5.613 lines (fewest-line map—minimal), about 18.000 the segment map derived from it (Fig. 1a). The result of the processing shows how the trend of the global integration index (the most important configurational parameter) is characterized by a strong centre-periphery gradient, with high values inside the ancient city and in their neighbors and much more segregated lines in the edge of the town. In this way

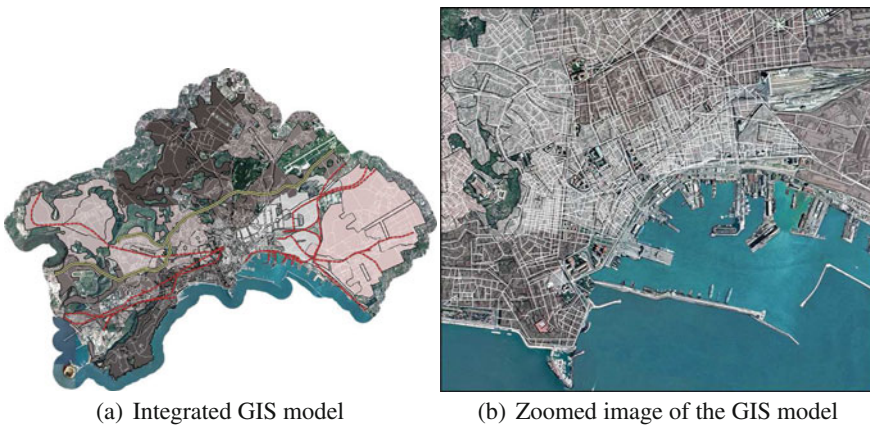


**Fig. 1** Urban network and Geo-statistics



it was easy to identify a single global centrality. Otherwise, for the identification of local centrality, on the basis of a distribution of local integration index (metric radius 400) flat with very few large peaks, we proceeded in a GIS environment according to the methodology first hatched (Fig. 1b). Were thus identified 47 local centrality distributed widely, but unevenly, with a clear concentration around the global main integrator and progressive thinning, although contradicted locally from a few exceptions. Having regard to the high number of lines, the transition from DepthMap to ESRI GIS environment has been done without ever dissociate geometric and information data. Filling out of the real estate values geo-database has been approached into two phases. The first one consisting in the georeferentiation of raster maps, containing the perimeters of homogenous price areas obtained by the Stock Exchange of Naples Real Estate. Then a second one of building the informative database containing market values and rental fees (both expressed per square meter of floor-type) for different types of real estate (residential, commercial, manufacturing, box) as well as support information such as local administration references. For the last step of the model, having already made uniform the work environment, the various information has been overlaid. At this stage, it was decided, through the specific capabilities of the GIS environment, to associate to each line of the configurational map its real estate value, attributed by reason of their geographical location. Even in this case it was necessary to manually correct the inconsistencies due to the overlap of urban routes. Likewise have been eliminated lines geographically belonging to areas devoid of price due to the impossibility to transact in them, such as the public or the public interest. To sum up, it is thus obtained a model that integrates on the same vector element all the information necessary for the development of the regression models first introduced, as well as a clear informative support able to allow a direct verification of the results in a way as simple and effective (Fig. 2).

The first of the two regression models, aimed at the study of phenomena on a global scale, relates the variation of the size of homogeneous price areas, seen as



**Fig. 2** Integrated spatial data approach

an indicator of sensitivity to the attractiveness/accessibility of urban space, with the distance from the global centrality. To obtain this result, the average size of homogeneous price areas was pushed on the centroid of each one of the 47 local centrality, considering the distance of 400 m along routes of public access as the criterion to determine which of the price area have to be associated to every local centrality. Associating the average surface to local centrality allows the regression to describe the real estate local market as an endogenous variable of the spatial model. Result that would have been unachievable if the distances of the individual areas from the global centrality had been purely considered, as it would not have taken into account the actual role played by the spatial configuration on local behavior and urban pattern use.

The second regression model, aimed at the study of local phenomena, relates real estate market trends and the configurational indexes, through the implementation of an ordinary least square model, using prices as explanatory variable and local integration index (metric radius 400) as response variable. A set of esteemed property values due to the distribution of configurational indexes have been so obtained. The difference between the estimated value and the recorded value has been calculated also, as well the deviation mapped.

## 5 Experimental Results

In the case of Naples, the correlation between the number of local centrality and the distance that separates them from the global centrality has a natural logarithmic trend through which it is possible to notice that as you move away from the center of the city, the decrease of high independent areas (high value of the local integration index—metric radius 400) will face more and more pronounced. This means the probability to be nigh to a local centrality decreases proceeding from the core to the outside of the city, in any direction.

It is valid, of course, the contrary assertion: whatever access to the city the chance to be close to a local centrality increases moving towards the core. The phenomenon occurs with great significance (Fig. 3a). Similarly, the number of local centralities increase as their own size get smaller, as shown by the stacked graph in Fig. 3b. This returns the city as composed of many small, fragmented areas.

Looking at homogeneous price areas, the correlation between their distribution in size and the distance from the main integrator clearly highlight a concentration near the global centrality. It is linear and sufficiently strong from the probabilistic point of view, being the coefficient of determination ( $R^2$ ), which is the square of the Pearson correlation coefficient, relatively high ( $R^2 = 0.7955$ ). It means that the 79.55% of the total variation in size of homogeneous price areas is explained by the relationship between it and the distance from the integration core (Fig. 3c). This result is congruent with the common interpretation in Space Syntax that the global centrality catalyzes the fragmentation of the areas according to a distance decay function [11]. This phenomenon is also congruent with the generative logic of the movement in

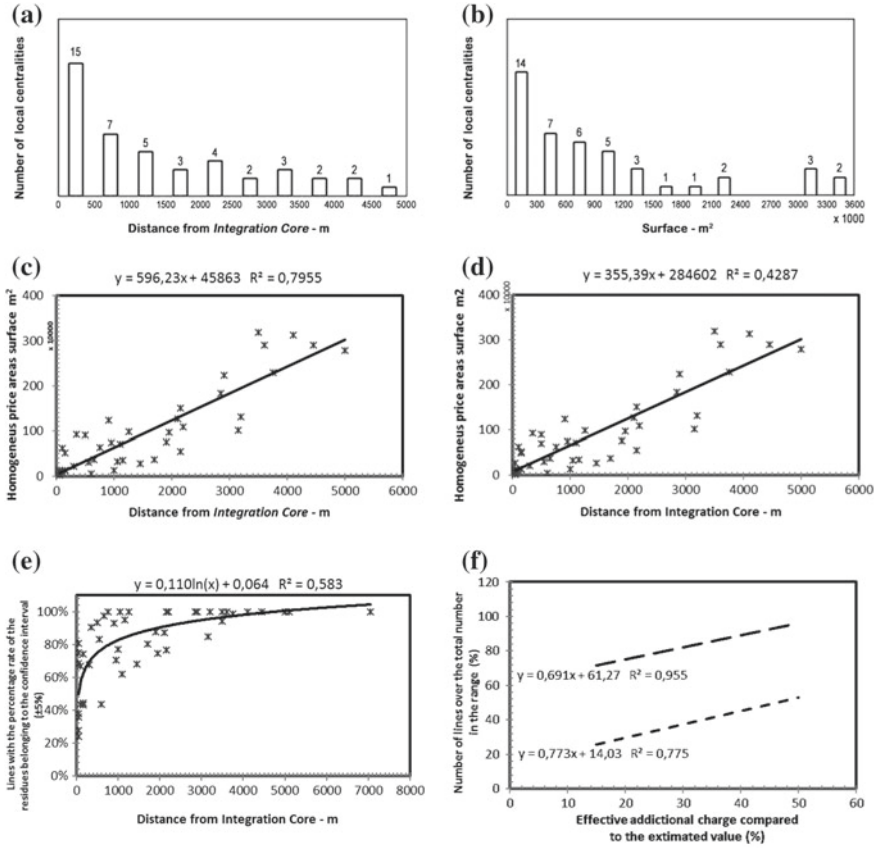


Fig. 3 Experimental results

urban areas, and especially with the experimental measurements which show that in the most integrated urban pattern its noticeable a logarithmic multiplier of pedestrian and vehicular flows [12]. Actually, a similar “area’s effect” arises looking at the variation of real estate rent: it acts logarithmically, due to the increasing of the distance from the main integrator. Global level outcomes have some local inconsistencies, especially related to two of the 47 identified local centralities. Inserting the latter in the regression model, in fact, the statistical quality drops drastically in the correlation between distance and size of price areas ( $R^2=0.4287$ ) (Fig. 3d). This is presumably due to the presence, in the influence sphere of such centralities, of an exceptional number of attractors, son of a fifty-year planning process of infrastructures and transports. This suggest the idea that the area’s effect is just a function of non-homogeneous distribution of the attractors as is amply demonstrated in the literature referring to other phenomena, consistently with the essential dynamics of urban spaces [13]. Locally, the comparative study between the arrangement of configura-

tional indexes (metric radius 400) and the property values has been formalized, as seen previously, through the use of the statistical tool of the Ordinary Least Squares (OLS). The result is the association with each line of an expected value of residential real estate. It shows very clearly the relationship which exists within the basin of each centrality, allowing us to extrapolate a series of evidence. Defined a confidence interval considered acceptable for the price volatility (5%) it is possible to see as the number of lines belonging to it grows logarithmically in moving from the integration core to the outside of the map (Fig. 3e). This situation confirms what was previously said, in global terms, as it shows that moving away from the core the sensitivity to accessibility and attractiveness decreases. This shows also that in areas where the presence of attractors is low, which are predominantly residential areas, the configuration is a correct indicator of the real estate market. If we analyze the centrality next to the core, the outlook is considerably more complicated. Specifically there is an interference between the purely local and global phenomena. This profoundly alters the interpretation of the market. Focusing on the lines that fall both in local centrality basins that in the core, the higher the percentage of difference between actual and expected market values linearly increases the number of lines that belong directly to the core or that are within a range of 20 m from it (Fig. 3f). This shows how the multiplier effect of commercial activities affect in a decisive manner the formation of real estate values, in compliance with what happens to pedestrian flows. The strength of this phenomenon blows over the market trend in mixed-use areas. It is the real phenomenon in place, which depends substantially on the variability of prices. This explains why just moving too little from the lines belonging to the core we record a substantial decrease of the differences between expected and recorded values. Local phenomena at smaller scale, certainly present, are not amenable to clear configurational contributions, at least in the context of the available data. Major experimental result are graphically summarized in Fig. 3.

## 6 Conclusion and Future Works

This paper examines the relationship between configurational indexes, both at global and local level, and between the distributions of real estate values, accounted as a proxy of location choices in the city. An empirical model was built in order to correlate endogenous locational measures (configurational indexes of angular segment integration and choice) varying analysis metric radius (6400 to 400 m), and real estate registered prices, gleaned over six months of market observations (more than 3400 normalized transactions during 2011 second semester). Configurational indexes have been shown to account for the distribution of movement intensity patterns for various transportation modes (pedestrian, cycle, motorized, and rail), and so they are good at culling spatial accessibility. Likewise, show their ability to seize the dynamics of the distribution of property values. At the global level, they offer a snapshot of the city consistent with the logic of the models AMM, or with a gradient distribution of prices decreasing from areas with high index values (CBD).

This gradient occurs in natural logarithmic terms, regardless of the local structure of the individual areas of homogeneous price. This is to say that the findings of classical economic geography are in fact confirmed as regards the interpretation of the phenomenon. Unlike these, however, the concept of centrality assumes endogenous nature. This then allows the model to capture the generative dynamics of the different price scenarios. It seems linked to the action of an *areal effect* that is distributed homogeneously in the surroundings of the local centrality (angular segment integration index metric radius 400) solely because of their distance from the global centrality (main integrator). Actually, we are investigating on the use of both objective aspects of urban space using an algorithmic approach and subjective issues related to the perception of communities which change the city, recurring to the formalism of ontology to represent urban elements both from a conceptual and topological point of view.

## References

1. Muth, R.: Cities and Housing. University of Chicago Press (1969)
2. Hillier, B., Hanson, J.: The Social Logic of Space. Cambridge University Press, UK (1984)
3. Hillier, B.: Space is the Machine. Cambridge University Press, UK (1996)
4. Cataldo, A., Pinto, V.D., Rinaldi, A.M.: Representing and sharing spatial knowledge using configurational ontology. *Int. J. Bus. Intell. Data Min* **10**(2), 123–151 (2015)
5. Wingo, L.: Transportation and Urban Land. Resources for the Future (1961)
6. Alonso, W.: Location and Land Use. Harvard University Press (1964)
7. Chiaradia, A., Hillier, B., Barnes, Y., Schwander, C.: Residential property value patterns. In: Proceedings of the 7th International Space Syntax Symposium, pp. 015:1–015:12. Stockholm, SVE (2009). KTH
8. Matthews, J., Turnbull, G.: Neighborhood street layout and property value: the interaction of accessibility and land use mix. *J. Real Estate Financ. Econ.* **35**, 111–141 (2007)
9. Cataldo, A., Cutini, V., Pinto, V.D., Rinaldi, A.M.: Subjectivity and objectivity in urban knowledge representation. In: Proceedings of the International Conference on Knowledge Discovery and Information Retrieval (KDIR-2014), pp. 411–417. Scitepress (2014)
10. Turner, A.: Angular analysis. In: Proceedings of the 3rd International Space Syntax Symposium, pp. 015:1–015:12 (2001)
11. B. Hillier. Centrality as a process. Accounting for attraction inequalities in deformed grids. In: Proceedings of the 2nd International Space Syntax Symposium, pp. 06.1–06.20 (1999)
12. Hillier, B., Penn, A., Hanson, J., Grajewski, T., Xu, J.: Natural movement: configuration and attraction in urban pedestrian movement. *Environ. Plan. B: Plan. Des.* **20**, 29–66 (1993)
13. Hillier, B.: The hidden geometry of deformed grids: or, why space syntax works, when it looks as though it shouldn't. *Environ. Plan. B: Plan. Des.* **26**, 169–191 (1999)

# A Partitioning Based Heuristic for a Variant of the Simple Pattern Minimality Problem

Maurizio Boccia, Adriano Masone, Antonio Sforza and Claudio Sterle

**Abstract** Logical Analysis of Data deals with the classification of huge data set by boolean formulas and their synthetic representation by ternary string, referred to as *patterns*. In this context, the simple pattern minimality problem (SPMP) arises. It consists in determining the minimum number of patterns “explaining” an initial data set of binary strings. This problem is equivalent to the minimum disjunctive normal form problem and, hence, it has been widely tackled by set covering based heuristic approaches. In this work, we describe and tackle a particular variant of the SPMP coming from an application arising in the car industry production field. The main difference with respect to SPMP tackled in literature resides in the fact that the determined patterns must be partitions and not covers of the initial binary string data set. The problem is solved by an effective and fast heuristic, tested on several large size instances coming from a real application.

**Keywords** Simple pattern minimality • Minimum disjunctive form • Optimal diversity management

---

M. Boccia  
Department of Engineering, University of Sannio, Piazza Roma, 21,  
82100 Benevento, Italy  
e-mail: maurizio.boccia@unisannio.it

A. Masone · A. Sforza · C. Sterle (✉)  
Department of Electrical Engineering and Information Technology,  
University “Federico II” of Naples, Via Claudio 21, 80125 Naples, Italy  
e-mail: claudio.sterle@unina.it

A. Masone  
e-mail: adriano.masone@unina.it

A. Sforza  
e-mail: sforza@unina.it

## 1 Introduction

By wiring we mean the positioning of conducting wires (*cables*) to make operative the electrical components of a device/system, connecting them to the control unit. A positioning, to be effective and efficient, should be made gathering together, in common paths, the largest number of wires. This definition can be obviously extended to the car industry, where *wiring* a vehicle means setting up all the cables to make operative all the electrical components required by a customer. More precisely, vehicles can be equipped with different standard options (installed by default) and extra options (installed on request). In other words, given a set of options chosen by a customer (i.e., a *demand*), wiring a vehicle means installing the required cables (*wiring configuration*) to make them operative.

Nowadays, the number of available options is very large (from 10 to about 50) and it is not going to decrease, since market trend pushes towards differentiation and highly customized products. Hence, even if a customer cannot require any combination of options because of wiring constraints, the number of possible demands is growing faster and getting huge. To give an idea of the real problem sizes, a vehicle with 20 options has 196,608 admissible requests. By wiring constraints, we mean several restrictions on the installation of two or more options together. Three main kinds of constraints must be considered:

1. **Incompatibility.** Two or more options are said to be incompatible if they cannot co-exist in the same demand. For example, right and left control for the audio car system cannot be installed together.
2. **Exclusivity.** Two or more options can be installed just if others are already present. For example, steering wheel controls are linked to the presence of car radio and door remote control is linked to central locking.
3. **Functional package.** Two or more options compose a modular system, hence they must be always positioned together, e.g. car radio is sold either with 4/6 loudspeakers and subwoofer or with a satellite navigator.

In this context, for a car industry, satisfying all the admissible demands just using exactly the related cables, would mean to have a specific wiring configuration for each of them. This is obviously impossible, since it would mean that a car industry should produce in advance and manage at the assembly line a huge number of wiring configurations. For this reason, car industries produce a limited number of opportunely chosen wiring configurations. Then, if a wiring configuration containing just the options of an admissible demand is not produced, the car industry substitutes it with a compatible (dominating) one. For the sake of the clarity, let us consider a small example. A configuration  $U$  of 3 options has to be installed:  $U = \{A, B, C\}$ . This configuration could be substituted by the dominating configuration  $V$ ,  $U \subseteq V$ , with the following 4 options:  $V = \{A, B, C, D\}$ . This substitution allows the car industry to overcome the problem of managing a great number of wiring configurations. However, on the other side, each time the car industry performs such substitution, it also sustains an additional wiring cost (extra-cost),

since it is giving “*as gift*” the cables for one or more options not demanded by the customer.

Hence the choice of the number and the kind of wiring configurations to produce should be done in order to minimize this extra-cost. This problem, referred in the literature to as optimal diversity management (*ODM*), has been introduced by Briant and Naddef [2] and formulated and solved as a  $p$ -median problem in Avella et al. [1]. The problem that is going to be tackled in this paper comes downstream the one just described. Indeed, given the set of options and related costs, the set of available wiring configurations at the assembly line and the set of admissible demands, the car industry wants to synthetically represent the sub-set of demands (partitions) assigned to each wiring configuration through the usage of ternary strings, referred to as *patterns*. Hence, a variant of the set partitioning problem arises, where the aim is defining the minimum number of patterns coherent with the available wiring configurations, covering all the admissible demands and maintaining the condition that each demand is assigned to the configuration covering it at the minimum cost. As will be better explained in the following section, this problem has been already defined in the field of the Logical Analysis of Data (LAD) first described in [4]. More precisely it can be considered as a particular variant of the simple pattern minimality problem [5, 6] which in turn, coincides with the minimum disjunctive normal form problem that is *NP-complete* [3].

The work is structured as follows: in Sect. 2 we explain the simple pattern minimality problem and the variant we tackle in this work; in Sect. 3 we propose a heuristic approach for our problem; finally, in Sect. 4, we will present several computational results.

## 2 Problem Description and Setting

In order to describe the problem, let us define the following sets:

- $O$  set of available options with the related installation cost  $c_o$ ;
- $WR$  set of wiring configurations, expressed as binary strings of  $|O|$  elements. The generic element  $WR_i(o)$  of a wiring configuration  $WR_i$ , assumes value 1, if  $WR_i$  contains the cable of option  $o$ , 0, otherwise
- $D$  set of admissible demands, expressed as binary strings of  $|O|$  elements. The generic element  $d_j(o)$  of a demand  $D_j$  assumes value 1, if the demand  $D_j$  contains option  $o$ , 0, otherwise
- $P$  set of patterns, expressed as ternary strings of  $|O|$  elements. The generic element  $p_k(o)$  of a pattern  $P_k$  assumes value 1, if the option  $o$  is in the pattern  $P_k$ ; 0, if the option  $o$  is not in  $P_k$ ; 2, if the option  $o$  can or cannot be in the pattern  $P_k$ . A pattern covers a demand just iff  $d_k(o) = p_k(o)$  for each  $o$  such that  $p_k(o) \in \{0, 1\}$ .

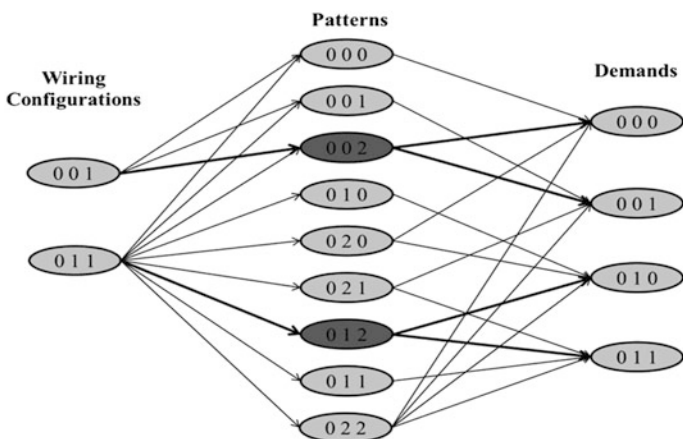


**Fig. 1** Patter example and related covered demands

$P_1$	0	1	0	0	0	2	0	2	1
$D_1$	0	1	0	0	0	0	0	0	1
$D_2$	0	1	0	0	0	0	0	1	0
$D_3$	0	1	0	0	0	1	0	0	0
$D_4$	0	1	0	0	0	1	0	1	1

To better explain the concept of *pattern*, let us consider the example of Fig. 1, where we report a pattern,  $P_1$  and related demands  $D_1, D_2, D_3$  and  $D_4$ .

In the *simple pattern minimality problem* (SPMP) the aim is determining the minimum number of patterns covering exactly the set of demands  $D$ , with no restriction on the generation of pattern (each demand can be covered by more than one pattern). The car industry problem tackled in this work is a particular variant of the SPMP, where the main difference is in the fact that the determined patterns must be partitions and not covers of the initial binary string data set. More precisely, it can be summarized as follows: (1) determining the minimum number of demand partitions; (2) generating the patterns representing the partitions, coherently with the available wiring configurations (patterns have to be a cover of the wiring configuration); (3) assigning each generated pattern to exactly one coherent wiring configuration, covering all the related demands at the minimum cost. For the sake of the clarity, we provide a graphical representation and explanation of the problem. Let us assume to have all possible patterns coherent with the available wiring configurations. Then the problem can be represented by a multi-level graph (Fig. 2), with wiring configurations, patterns and demands on first, second and third level, respectively. The arrows show the “coherency”, i.e. the possible assignments,



**Fig. 2** Multi-level graph representation of the problem

between wiring configurations and patterns, and, patterns and demands. We highlight, by thicker lines, a feasible solution of the problem where each demand is covered by a single pattern and each pattern is assigned to a single wiring configuration covering it at the minimum cost, i.e., the extra cost for all the demands paid when replacing a configuration with a richer one, is minimized.

Two issues must be further discussed about the structure of represented feasible solution for the problem. First one is that the demand partitioning is guaranteed by the fact that the selected patterns differ for at least one element (the second one) which assumes value 0 in *Pattern 3* and value 1 in *Pattern 7*. This means that the two patterns have no demands in common. This would not occur if, for example, the solution was made by *Pattern 3* together with *Pattern 9*, since first and second demands would have been covered by both patterns. Second issue concerns the tradeoff between the extra cost and patterns (partitions) minimization. Indeed, if the *Pattern 9* was the only one to be selected to represent the four demands, it would have been associated to the second wiring configuration, the only one coherent with this pattern. Given that second wiring configuration is the most expensive one, this would imply a higher extra-cost for the car industry with respect to the highlighted solution, since demands 1 and 2 are not covered by the cheapest wiring configuration.

### 3 Partitioning Based Heuristic

The basic idea of the proposed solving heuristic is the decomposition of the main set partitioning problem in more set partitioning sub-problems, one for each wiring configuration. The algorithm starts from the complete knowledge of the list of admissible customer demands  $D$  but does not enumerate all the possible patterns associated to the available wiring configurations. It develops in two phases:

- **Preprocessing: Demand-Wiring configuration assignment.** Define the sub-lists of demands assigned to each wiring configuration. Each demand has to be assigned to the wiring configuration covering it at the minimum cost.
- **Core: Pattern generation.** Determine the patterns associated to each wiring configuration with no multiple assignment of demands.

#### 3.1 Preprocessing: Demand-Wiring Configuration Assignment

In this phase a simple minimum coverage cost problem is solved. The steps to be performed are two:

1. Order the  $WR_i$  for increasing cost values ( $\sum_{o \in O} c_o WR_i(o)$ ),
2. Assign each demand to the wiring configuration covering it at minimum cost.

The assignment at the minimum cost between demands and wiring configuration is obtained performing a *bit-to-bit OR* operation between the elements of the binary string representing the wiring configuration and the binary string representing the demand. The minimum cost assignment is guaranteed by the fact that the *OR* operation is done for each demand and all possible wiring configuration starting from the top of the ordered list. In this way demand sub-lists are generated for each wiring configuration.

### 3.2 Core: Pattern Determination

Given the result of the preprocessing phase, a pattern list is generated by successive aggregation of demands assigned to a wiring configuration  $WR_i$ . In particular, the *pattern determination* phase requires the following operations, where  $\mathbf{D}$  and  $\mathbf{P}$  denote the demand and the pattern list respectively:

Step 0. Order the list  $\mathbf{D}$  for increasing values of the number of bits equal to 1;

$$\text{set } P = [D_1], j = 1 \text{ and } i = 2.$$

Step 1. Evaluate the “bit-to-bit difference” between  $D_i$  and  $D_j$ , for each  $j \leq i$

- if there is just one bit of difference between  $D_i$  and  $D_j$ , then change value of the difference bit to 2 in the pattern  $P_k$  covering  $D_j$
- else insert  $D_i$  in  $\mathbf{P}$  and set  $j = j + 1$ .

Step 2. Set  $i = i + 1$ .

Step 3. If  $i \leq |D|$  return to Step 2 else STOP

In order to better explain this phase, let us consider a small example (Fig. 3), where a wiring configuration and related list of ordered demands are given.

The first three demands have the same number of elements equal to 1. This means that, performing a bit to bit difference between two of them the result will always have more than one bit of difference. Hence the first three demands are directly inserted in the pattern list, which, after three iterations of the aggregation procedure will have the following structure:

$$P_1 = 000000100 \quad P_2 = 000001000 \quad P_3 = 000010000$$

**Fig. 3** Wiring configuration and associated ordered demand sub-list

	O <sub>1</sub>	O <sub>2</sub>	O <sub>3</sub>	O <sub>4</sub>	O <sub>5</sub>	O <sub>6</sub>	O <sub>7</sub>	O <sub>8</sub>	O <sub>9</sub>
WR <sub>1</sub>	1	1	0	0	1	1	1	0	1
	O <sub>1</sub>	O <sub>2</sub>	O <sub>3</sub>	O <sub>4</sub>	O <sub>5</sub>	O <sub>6</sub>	O <sub>7</sub>	O <sub>8</sub>	O <sub>9</sub>
D <sub>1</sub>	0	0	0	0	0	0	1	0	0
D <sub>2</sub>	0	0	0	0	0	1	0	0	0
D <sub>3</sub>	0	0	0	0	1	0	0	0	0
D <sub>7</sub>	0	0	0	0	0	0	1	0	1
D <sub>5</sub>	0	0	0	0	0	1	1	0	0
D <sub>8</sub>	1	0	0	0	0	1	0	0	0
D <sub>4</sub>	0	1	0	0	1	0	0	0	0
D <sub>6</sub>	0	0	0	0	1	0	1	0	1

Let us consider now the fourth demand  $D_7$  with two bits equal to 1, and perform the bit-to-bit difference between this demand and the previous ones:

$$\begin{aligned}
 D_7 &= 000000101 \\
 D_1 &= \underline{000000100} \\
 Diff. &= 000000001
 \end{aligned}$$

There is just one bit of difference between  $D_7$  and  $D_1$  and it is in correspondence of element  $O_9$ . For this reason, the two demands can be represented by the same pattern, where the value of element  $O_9$  of the pattern to which  $D_1$  is assigned to is changed to 2:

$$P_1 = 000000102$$

Repeating the same procedure for fifth demand  $D_5$ , we will obtain that  $P_1$  has to be modified as follows:

$$P_1 = 000002102$$

Concerning this modification, it is important to underline the fact that  $P_1$  has been generated aggregating just three demands, but it synthetically represents four demands, since it also includes the demand  $D^* = [000001101]$ . This occurrence is acceptable for the car industry firm since: if this demand is admissible, then it would be in any case assigned to the same wiring configuration, which is the one covering it at minimum cost; if it is not admissible, even if it is covered, it cannot be required by any customer.

Repeating the same procedure for all the demands assigned to the  $WR_1$  wiring configuration the following patterns will be obtained:

$$P_1 = 000002102 \quad P_2 = 200001000 \quad P_3 = 020010000$$

As it can be noticed, in this way, instead of storing for  $WR_I$  wiring configuration the complete set of covered demands, they are synthetically represented by the usage of three patterns.

### 3.3 Algorithm Refinement

The above described algorithm has a weakness point which has to be taken into account. The most time consuming part is the demand aggregation phase, since in this phase we evaluate each current demand with all the previous ones with a small number of bits equal to 1. This operation, given the huge amount of possible demands, can be very heavy from the computation time point of view. For this reason, in order to significantly reduce it, a small modification has been performed. Instead of evaluating at each iteration the bit to bit difference of the current demand with all the previous ones, we evaluate the bit to bit difference between the current demand and all the patterns generated until that moment. This difference is performed without considering the element of the patterns which have already been set to 2, since this means that they can assume both values, 0 or 1. This adjustment significantly reduce the number of bit to bit operations that have to be performed. On the other side, it requires an additional control in order to avoid the possibility of having a demand covered by more than one pattern. In fact, let us consider a situation where, during the aggregation phase, the following two patterns have been generated:

$$P_1 = 010120101 \quad P_2 = 012012101$$

Suppose now that the following demand has to be taken into account  $D^* = [011111101]$ . This demand has more than one bit of difference if compared with  $P_1$  and just one bit of difference if compared with  $P_2$ . Hence, it can be aggregated to  $P_2$  setting the value of fourth element to 2. But if we perform such modification of pattern  $P_2$ , the demand  $\hat{D} = [010100101]$  could be covered by both patterns, violating single assignment constraints. For this reason, we have to add a control which, each time a pattern should be modified, checks if this modification produces a pattern which completely covers another existing one. If this happen the aggregation is not possible and the current demand is introduced in the pattern list as a new pattern, otherwise the aggregation is performed.

## 4 Instance Details and Computational Results

The proposed algorithm has been experienced on six test cases provided by the car industry. The details of these set cases are summarized in the Table 1: first column reports the instance identifier; second column, the number of available options;

**Table 1** Instance details

Test cases	# options	# <i>WR</i>	# combinations	# demands
1	20	61	1,048,576	196,608
2	26	81	67,108,864	1,179,648
3	33	37	8,589,934,592	393,600
4	36	50	68,719,476,736	111,476,736
5	38	24	274,877,906,944	215,239,680
6	47	25	140,737,488,355,328	2,179,989,504

third column, the number of wiring configurations; fourth column, the number of possible combinations of options; fifth column, the number of admissible customer demands. As said above it is important to note that the number of admissible demands is much lower than the one of the possible combinations and this is due to the wiring constraints. This motivates the choice of using as input data of the algorithm the complete list of the admissible demands.

Proposed algorithm has been run on a Windows XP 64 bit Intel(R) Core 2 Duo CPU T9300–2,50 GHz—RAM 4,00 GB. It has been compared in terms of computation time and number of generated patterns with solutions already used by the car industry, reported in third and fourth column of Table 2. It is easy to see that the main criticalities of the available solution regard mainly medium and large size instances (the last three test cases), for which the computation time started to significantly increase. Moreover, no solution was available for test case 6. In fifth and sixth column of Table 2, the results of the proposed algorithm are reported. It can be observed that results on the first three test cases are very similar in terms of number of patterns and computation time, whereas significant improvement have been obtained on the last three instances on both quality of solution and computation time. More precisely the number of generated patterns are about a half of the available solutions and computation time are less than half an hour (instead of several hours and days) for test case 4 and 5. A solution for test case 6 has been found in about 3 h and 30 min.

**Table 2** Computational results

Test cases	# options	# patterns (avail. sol.)	Time (avail. sol.)	# patterns (alg. sol.)	Time (alg. sol.)
1	20	66	<1 m	66	<1 m
2	26	131	<1 m	131	<1 m
3	30	250	2 m	250	<2 m
4	33	685	9 h 32 m	306	25 m
5	38	992	19 d 21 h 27 m	411	29 m
6	47	–	–	496	3 h 30 m

## 5 Conclusions

In this paper a particular variant of the SPMP, arising in the car industry, has been tackled. Given the sizes of the real cases to be treated and the need to have good solutions in reasonable computation time, the problem has been solved by an ad hoc heuristic approach. The proposed algorithm develops in two phases and is based on the decomposition of the main set partitioning problem in more sub-problems, one for each available wiring configuration. All the test instances have been solved, including also one hard large instance containing 47 options.

## References

1. Avella, P., Boccia, M., Di Martino, C., Oliviero, G., Sforza, A., Vasilev, I.: A decomposition approach for a very large scale optimal diversity management problem. *4OR: Q. J. Oper. Res.* **3** (1), 23–37 (2005)
2. Briant, O., Naddef, D.: The optimal diversity management problem. *Oper. Res.* **52**(4), 515–526 (2004)
3. Garey, M.R., Johnson, D.S.: *Computers and intractability. A guide to the theory of NP-completeness. A Series of Books in the Mathematical Sciences.* W. H. Freeman and Company, San Francisco (1979)
4. Hammer, P.L.: Partially defined boolean functions and cause-effect relationships. In: *Lecture at the International Conference on Multi-attribute Decision Making Via OR-Based Expert Systems.* University of Passau, Passau, Germany (1986)
5. Lancia, G., Serafini P.: A set-covering approach with column generation for parsimony haplotyping. *JOC: J. Comput.* **21**(1), 151–166 (2009)
6. Lancia, G., Serafini P.: The complexity of some pattern problems in the logical analysis of large genomic data sets. *Sets.* In: Ortuño, F., Rojas, I. (eds.), *Bioinformatics and Biomedical Engineering. IWBBIO 2016. Lecture Notes in Computer Science*, vol. 9656. Springer, Cham

**Part III**  
**Health Care**



# Patient–Centred Objectives as an Alternative to Maximum Utilisation: Comparing Surgical Case Solutions

Roberto Aringhieri and Davide Duma

**Abstract** Operating Room (OR) planning and scheduling is a research topic widely discussed in the literature, in which several performance criteria have been proposed to evaluate the OR planning decisions. Although the OR utilisation is the leading objective, from research experiences, long waiting lists lead to a satisfactory filling of ORs even fixing other objectives. In this paper we analyse the impact on OR utilisation of two patient–centred objectives: the waiting time minimisation and the workload balance. In the former the most commonly used patient–centred criterion is taken into account, while the latter leads to a smooth stay bed occupancies determining a smooth workload in the ward and, by consequence, an improved quality of care provided to patients. To the best of our knowledge, a comparison of the planning determined by these criteria is not yet available in literature.

**Keywords** Surgery process scheduling · Patient–centred · Objective functions Comparison

## 1 Introduction

Operating Room (OR) planning and scheduling is a research topic widely discussed in the literature [9, 12, 21]. At the operational decision level [23], the problem arising is also called “surgery process scheduling” of elective patients, which usually consists in (i) selecting patients from an usually long waiting list and assigning them to a specific OR session (i.e., an operating room on a specific day) over a planning horizon [3, 17, 20], and (ii) determining the precise sequence of surgical procedures and the allocation of resources for each OR session [13, 18]. A further

---

R. Aringhieri · D. Duma (✉)  
Dipartimento di Informatica, Università degli Studi di Torino,  
Corso Svizzera 185, 10149 Torino, Italy  
e-mail: davide.duma@unito.it

R. Aringhieri  
e-mail: roberto.aringhieri@unito.it

problem was recently introduced: the Real Time Management (RTM) of operating rooms [2, 10] is the decision problem arising during the fulfillment of the surgery process scheduling, that is the problem of supervising the execution of such a schedule and, in case of delays, to take the more rational decision regarding the surgery cancellation or the overtime assignment. The RTM with also non-elective patients is studied in [11] in which a competitive analysis of the online algorithms is also discussed. Such problems are further challenged by the inherent stochasticity of their main parameters, such as the surgery duration, the length of stay and the arrival of non-elective patients [1, 6, 14].

Several performance criteria have been reported to evaluate the OR planning decisions [9]. Usually, the maximisation of the OR utilisation is the most important criterion, since ORs are the largest cost and revenue centre of hospitals [12]. However, long waiting lists could lead to a satisfactory OR utilisation due to the broader variety of surgery durations that can be selected adopting other objectives. Note that long waiting lists is a common situation in many hospitals belonging to publicly funded health care systems. Taking into account a patient-centred perspective, one of the most commonly used criterion is the waiting time, which is addressed weighting the patients according to the time elapsed from the referral day [3, 22, 24]. Conversely, the workload balance criteria is designed for workers in the wards, leading to a smooth—without peaks—stay bed occupancies that determines a smooth workload and, by consequence, an improved quality of care provided to patients [4, 5, 19].

The main contribution of this paper is the comparison of the surgical planning determined by these criteria. To the best of our knowledge, some attempts are available in the literature but they are applied to different operative contexts [7, 16] or taking into account different goals [8]. To perform the comparison, we will consider the surgical case assignment problem under the two criteria. Note that the surgical case assignment problem has been proved to be  $\mathbb{N}^{\mathbb{P}}$ -hard in [3]. We compute the solutions of such problems considering a set of instances generated by using the instance generator proposed in [15].

## 2 Mathematical Formulations

In this section, we propose simple integer linear programs to model the surgical case assignment problem under the two criteria considered, that is the minimisation of the waiting times and the workload balance. In the following we will consider a single specialty with a large number of stay beds and a long waiting list of patients. These assumptions would avoid or limit the impact on the solutions of exogenous factors. The time horizon is one week composed of five operating days (from Monday to Friday).

Let  $I$ ,  $K$  and  $T$  be respectively the sets of patients, operating rooms and working days of the planning horizon, each indexed by the corresponding letter,  $i$ ,  $k$  and  $t$ . Note that each OR session in the planning horizon is uniquely defined by a pair of

indices  $(k, t)$ . We denote by  $s_{kt}$  the time capacity of the OR session  $(k, t)$ . For each patient  $i \in I$ , the following information are given: the waiting time  $w_i$ , expressed in days and computed from the referral day; the expected duration of the surgery  $p_i$ , expressed in minutes; the expected Length of Stay (LOS)  $\mu_i$ , expressed in days. Note that we assume that the value of  $\mu_i$  includes also the operating day, and that  $\mu_i \leq 3$  to avoid the case in which a bed is used by a patient operated on the previous week.

Let  $x_{ikt}$  be the binary decision variable that models the assignment of the patient  $i$  to the OR session  $(k, t)$  ( $x_{ikt} = 1$ ), or not ( $x_{ikt} = 0$ ).

$$M_1 : \quad \max \quad z_{\text{waiting}} = \sum_{i \in I} w_i \sum_{k \in K} \sum_{t \in T} x_{ikt} \quad (1a)$$

$$\text{s.t.} \quad \sum_{k \in K} \sum_{t \in T} x_{ikt} \leq 1, \quad i \in I \quad (1b)$$

$$\sum_{i \in I} p_i x_{ikt} \leq s_{kt}, \quad k \in K, t \in T \quad (1c)$$

The model  $M_1$  minimises the waiting time. The selection of the patients from the waiting list and their assignment to OR sessions is modelled by the constraints (1b) and (1c): constraints (1b) state that a patient can be scheduled at most once, while constraints (1c) impose that the sum of the surgery times of the patients scheduled in each OR session  $(k, t)$  may not exceed the time capacity  $s_{kt}$ . The objective function (1a) leads to a solution made of those patients having longest waiting time at the moment of planning. In other words, the model would favour those patients with longest waiting time instead of those with shorter ones.

$$M_2 : \quad \max \quad z_{\text{balance}} = y \quad (2a)$$

$$\text{s.t.} \quad \sum_{k \in K} \sum_{t \in T} x_{ikt} \leq 1, \quad i \in I \quad (2b)$$

$$\sum_{i \in I} p_i x_{ikt} \leq s_{kt}, \quad k \in K, t \in T \quad (2c)$$

$$\sum_{k \in K} x_{ikt} = y_{it}, \quad i \in I, t \in T \quad (2d)$$

$$\sum_{h=t}^{\max\{t+\mu_i, \ell''\}} z_{ih} \geq \mu_i y_{it}, \quad i \in I, t \in T \quad (2e)$$

$$\sum_{h=\min\{t-\mu_i, \ell'\}}^t y_{ih} \geq z_{it}, \quad i \in I, t \in T \quad (2f)$$

$$\sum_{i \in I} z_{it} \geq y, \quad t \in T \quad (2g)$$

The model  $M_2$  balances the workload. As in the previous model, constraints (2b) and (2c) represent the selection of the patients from the waiting list and their assignment to OR sessions. Constraints (2d) define the value of the auxiliary variables  $y_{it}$ ,

which is equal to 1 when the patient  $i$  is operated on the day  $t$ , 0 otherwise. Such a variable is then used to count the number of stay beds used each day  $t$  of the planning horizon  $T$  by fixing the value of the auxiliary variables  $z_{it}$ , which is equal to 1 when the patient  $i$  take up a stay bed during the day  $t$ , 0 otherwise through the constraints (2e) and (2f), in which the parameters  $\ell'$  and  $\ell''$  represent the first and the last working days in  $T$ , respectively. In order to model the work balance, we adopt a bottleneck approach: the objective function (2a) seeks to maximise the number of busy stay beds during the day with the minimal bed usage (constraints (2g)).

### 3 Comparison

In our comparison we consider a benchmark set of 11 instances whose case mix has been generated using [15], rounding the value to the nearest integer to obtain  $p_i$ . With a different case mix, each instance is composed of 300 patients whose  $w_i$  and  $\mu_i$  are uniformly distributed in  $[1, 365]$  and  $\{1, 2, 3\}$ , respectively. The operating theatre is composed of 4 ORs for a total operating time of 9600 min. The two models are implemented and solved using Cplex Optimization Studio 12.5 with a running time limit set to 300 s.

Table 1 reports the results of the comparison. For each instances (denoted as in [15]), the running time, the gap, the number  $N$  of operated patients, the total utilisation  $D$  of the ORs (minutes), and the average  $p_i$  of the operated patients are reported for both models  $M_1$  and  $M_2$ .

A first remark concerns the running time: the solution of the model  $M_1$  requires a larger computational effort than the model  $M_2$  even if the average solution gap of the model  $M_1$  is still quite limited. We observe that the solutions of the model  $M_1$  has

**Table 1** Comparing  $M_1$  and  $M_2$  solutions: main parameters

	Time		Gap		$N$		$D$		Avg. $p_i$	
	$M_1$	$M_2$	$M_1$ (%)	$M_2$ (%)	$M_1$	$M_2$	$M_1$	$M_2$	$M_1$	$M_2$
CHI	300.05	299.91	1.27	1.27	177	190	9598	9308	54.2	49.0
ENT	300.06	0.67	0.16	0.00	157	156	9573	9003	61.0	57.7
EYE	300.14	0.48	0.16	0.00	194	182	9583	9007	49.4	49.5
GYN	300.14	0.69	0.12	0.00	162	162	9576	8999	59.1	55.5
MIX	300.13	0.69	0.07	0.00	203	211	9579	9279	47.2	44.0
NEU	300.08	1.22	0.03	0.00	254	266	9588	9413	37.7	35.4
ONC	300.08	0.73	0.14	0.00	168	162	9582	9207	57.0	56.8
ORT	300.06	1.09	0.17	0.00	164	167	9577	8997	58.4	53.9
PLA	300.09	0.8	0.30	0.00	146	150	9558	9114	65.5	60.8
THO	300.06	161.78	0.20	0.00	162	182	9571	9506	59.1	52.2
URO	300.06	0.42	0.26	0.00	170	172	9577	9097	56.3	52.9

always a larger average OR utilisation than  $M_2$ , that is 99.8 and 95.6% over the total operating time, respectively. We observe that the OR utilisation of  $M_1$  is always close to 9600 min because of the proxy effect of its objective function. Quite surprisingly, the number  $N$  of operated patients not complies with the OR utilisation: those of the model  $M_2$  is larger than those in  $M_1$  in 7 instances over 11. Note that there is not a correlation between the value of  $N$  and  $D$ : actually, for the instance ORT, the difference between the values of  $D$  is 580 in favour of  $M_1$  (the largest one) while the difference between the values of  $N$  is 3 in favour of  $M_2$ ; for instances THO, the difference between the values of  $D$  is 65 in favour of  $M_1$  (the smallest one) while the difference between the values of  $N$  is 20 in favour of  $M_2$  (the largest one).

Table 2 reports the number of occupied stay beds for each day of the planning horizon. Further, the columns with bold values report the difference between the maximum and the minimum of those values. The reported results show the impact of not considering the balance of the workload in the solution of model  $M_1$ : while in  $M_2$  the average difference between the maximum and the minimum of occupied stay beds is 1.8, the same value in  $M_1$  is 43.7. We remark that the LOS of each patient can have an impact: actually, the sum of the beds over the time horizon is greater than the value of  $N$ . In the following tables we report the results for the instance CHI and THO varying the LOS distribution as follows: in the scenario 1, the 60% of the patients have  $\mu_i = 1$ , the 20% have  $\mu_i = 2$ , and the remaining 20% have  $\mu_i = 3$ ; on the contrary, in the scenario 2, the 60% of the patients have  $\mu_i = 3$ , the 20% have  $\mu_i = 2$ , and the remaining 20% have  $\mu_i = 1$ .

Table 3 reports the same type of results of those reported in Table 1 but only for the instances CHI and THO. While the number of operated patients is almost the same for model  $M_1$  in both scenarios, for model  $M_2$  the value of  $N$  increases in the scenario 2, that is when the patients with maximum LOS are the majority.

**Table 2** Comparing  $M_1$  and  $M_2$  solutions: occupied stay beds

	Model $M_1$						Model $M_2$					
	1	2	3	4	5		1	2	3	4	5	
CHI	41	73	70	70	58	<b>32</b>	79	79	79	79	79	<b>0</b>
ENT	40	66	70	50	55	<b>30</b>	60	64	65	60	60	<b>5</b>
EYE	40	66	91	72	68	<b>51</b>	65	66	66	65	65	<b>1</b>
GYN	33	61	78	66	56	<b>45</b>	59	65	59	59	59	<b>6</b>
MIX	43	77	81	90	81	<b>47</b>	85	88	85	85	85	<b>3</b>
NEU	49	126	116	83	96	<b>77</b>	115	115	115	115	115	<b>0</b>
ONC	34	57	67	67	69	<b>35</b>	57	57	59	57	57	<b>2</b>
ORT	34	56	78	65	62	<b>44</b>	68	69	68	68	68	<b>1</b>
PLA	29	51	57	56	51	<b>28</b>	56	56	56	56	57	<b>1</b>
THO	30	66	79	58	57	<b>49</b>	79	79	79	79	79	<b>0</b>
URO	32	61	65	61	75	<b>43</b>	63	63	63	63	64	<b>1</b>

**Table 3** Comparing  $M_1$  and  $M_2$  solutions: impact on the solution varying the LOS of the patients

	Time		Gap		$N$		$D$		Avg. $p_i$	
	$M_1$	$M_2$	$M_1$ (%)	$M_2$ (%)	$M_1$	$M_2$	$M_1$	$M_2$	$M_1$	$M_2$
Scenario 1										
CHI	300.08	1.92	0.07	0.00	176	181	9588	9360	54.5	51.7
THO	300.06	100.08	0.20	0.00	162	169	9571	9567	59.1	56.6
Scenario 2										
CHI	300.05	299.89	0.06	1.19	177	193	9590	9361	54.2	48.5
THO	300.08	2.25	0.17	0.00	161	176	9571	9232	59.4	52.5

**Table 4** Comparing  $M_1$  and  $M_2$  solutions: occupied stay beds varying the LOS of the patients

	Model $M_1$						Model $M_2$					
	1	2	3	4	5		1	2	3	4	5	
Scenario 1												
CHI	41	61	58	58	47	<b>20</b>	71	71	71	71	71	<b>0</b>
THO	28	45	55	51	58	<b>30</b>	62	62	62	62	62	<b>0</b>
Scenario 2												
CHI	41	78	89	82	68	<b>48</b>	84	84	103	84	84	<b>19</b>
THO	28	68	81	66	77	<b>53</b>	74	74	108	74	74	<b>34</b>

Table 4 reports the same type of results of those reported in Table 2 but only for the instances CHI and THO. For the scenario 1, the results confirm the previous remarks. On the contrary, the results for the scenario 2 show a significant increment of the difference between the maximum and the minimum of occupied stay beds for model  $M_2$ : actually, such a difference is due to a peak in the third day while the other days are balanced; such a peak is due to the larger number of patients with  $\mu_i = 3$ .

## 4 Conclusions

We proposed and analysed two integer linear programming models for the surgical case assignment problem using patient-centred objectives as an alternative to the maximum OR utilisation.

Quantitative analysis confirmed the ability of the two models to ensure a high level of OR utilisation dealing with long waiting lists. The two criteria provided different results. The minimisation of the waiting times is a fairness criterion among patients that allowed us to have an OR utilisation close to 100% in all cases. Conversely, the workload balance is a criterion to have a smooth workload along the week, which has been able to schedule a high number of patients in most cases.

The results of the model  $M_1$  showed that waiting time minimisation should be a proxy of the OR utilisation maximisation, when the waiting list is quite long. From this perspective, the most considerable and counter-intuitive result is the non-compliance between the OR utilisation and the number of the planned patients, as reported in Table 1.

A further work can be the study of the impact of these criteria when they are adopted over time and if used concurrently with other optimization modules, such as the RTM of operating rooms.

**Acknowledgements** The authors would thank the student Rita Iacono for running part of the computational tests. The authors would also thank Greanne Leeftink and Erwin W. Hans for providing the instance generators and the manuscript of their submitted paper [15].

## References

1. Addis, B., Carello, G., Grosso, A., Tånfani, E.: Operating room scheduling and rescheduling: a rolling horizon approach. *Flex. Serv. Manuf. J.* **28**(1–2), 206–232 (2016)
2. Aringhieri, R., Duma, D.: The optimization of a surgical clinical pathway. In: Ören, M., et al (eds.) *Simulation and Modeling Methodologies, Technologies and Applications. Advances in Intelligent Systems and Computing*, vol. 402, pp. 313–331. Springer (2016)
3. Aringhieri, R., Landa, P., Soriano, P., Tånfani, E., Testi, A.: A two level metaheuristic for the operating room scheduling and assignment problem. *Comput. Oper. Res.* **54**, 21–34 (2015)
4. Aringhieri, R., Landa, P., Tanfani, E.: Assigning surgery cases to operating rooms: a VNS approach for leveling ward beds occupancies. In: *The 3rd International Conference on Variable Neighborhood Search (VNS'14). Electronic Notes in Discrete Mathematics*, pp. 173–180 (2015)
5. Beliën, J., Demeulemeester, E., Cardoen, B.: Building cyclic master surgery schedules with levelled resulting bed occupancy: a case study. *Eur. J. Oper. Res.* **176**, 1185–1204 (2007)
6. Bruni, M., Beraldi, P., Conforti, D.: A stochastic programming approach for operating theatre scheduling under uncertainty. *IMA J. Manag. Math.* **26**(1), 99–119 (2015)
7. Cappanera, P., Visintin, F., Banditori, C.: Comparing resource balancing criteria in master surgical scheduling: a combined optimisation-simulation approach. *Int. J. Prod. Econ.* **158**, 179–196 (2014). <https://doi.org/10.1016/j.ijpe.2014.08.002>
8. Cappanera, P., Visintin, F., Banditori, C.: Addressing conflicting stakeholders priorities in surgical scheduling by goal programming. *Flex. Serv. Manuf. J.* 1–20 (2016). <https://doi.org/10.1007/s10696-016-9255-5>
9. Cardoen, B., Demeulemeester, E., Beliën, J.: Operating room planning and scheduling: a literature review. *Eur. J. Oper. Res.* **201**, 921–932 (2010)
10. Duma, D., Aringhieri, R.: An online optimization approach for the real time management of operating rooms. *Oper. Res. Health Care* **7**, 40–51 (2015)
11. Duma, D., Aringhieri, R.: The real time management of operating rooms. In: Kahraman, C., Topcu, I. (eds.) *Operations Research Applications in Health Care Management. International Series in Operations Research & Management Science*. Springer (2017)
12. Guerriero, F., Guido, R.: Operational research in the management of the operating theatre: a survey. *Health Care Manag. Sci.* **14**, 89–114 (2011)
13. Herring, W., Herrmann, J.: The single-day surgery scheduling problem: sequential decision-making and threshold-based heuristics. *OR Spectrum* **34**, 429–459 (2012)
14. Landa, P., Aringhieri, R., Soriano, P., Tånfani, E., Testi, A.: A hybrid optimization algorithm for surgeries scheduling. *Oper. Res. Health Care* **8**, 103–114 (2016)

15. Leefink, A.G., Hans, E.W.: Development of a benchmark set and instance classification system for surgery scheduling. *J. Sched.* (2017). <https://doi.org/10.1007/s10951-017-0539-8>
16. Li, X., Rafaliya, N., Baki, M.F., Chaouch, B.A.: Scheduling elective surgeries: the tradeoff among bed capacity, waiting patients and operating room utilization using goal programming. *Health Care Manag. Sci.* **20**(1), 33–54 (2017)
17. Marques, I., Captivo, M., Pato, M.: An integer programming approach to elective surgery scheduling. *OR Spectrum* **34**, 407–427 (2012)
18. Meskens, N., Duvivier, D., Hanset, A.: Multi-objective operating room scheduling considering desiderata of the surgical teams. *Decis. Support Syst.* **55**, 650–659 (2013)
19. van Oostrum, J., van Houdenhoven, M., Hurink, J., Hans, E., Wullink, G., Kazemier, G.: A master surgical scheduling approach for cyclic scheduling in operating room departments. *OR Spectrum* **30**, 355–374 (2008)
20. Rizk, C., Arnaout, J.: ACO for the surgical cases assignment problem. *J. Med. Syst.* **36**, 1191–1199 (2012)
21. Samudra, M., Van Riet, C., Demeulemeester, E., Cardoen, B., Vansteenkiste, N., Rademakers, F.: Scheduling operating rooms: achievements, challenges and pitfalls. *J. Sched.* **19**(5), 493–525 (2016). <https://doi.org/10.1007/s10951-016-0489-6>
22. Tànfani, E., Testi, A.: A pre-assignment heuristic algorithm for the master surgical schedule problem (MSSP). *Ann. Oper. Res.* **178**(1), 105–119 (2010)
23. Testi, A., Tànfani, E., Torre, G.: A three-phase approach for operating theatre schedules. *Health Care Manag. Sci.* **10**, 163–172 (2007)
24. Valente, R., Testi, A., Tànfani, E., Fato, M., Porro, I., Santo, M., Santori, G., Torre, G., Ansaldo, G.: A model to prioritize access to elective surgery on the base of clinical urgency and waiting time. *BMC, Health Serv. Res. Ann. Oper. Res.* **9**(1) (2009)



# A Hierarchical Multi-objective Optimisation Model for Bed Levelling and Patient Priority Maximisation

Roberto Aringhieri, Paolo Landa and Simona Mancini

**Abstract** Operating Rooms (ORs) are one of the highest cost drivers of hospital budget and one of the highest source of income. Several performance criteria have been reported to lead and to evaluate the OR planning decisions. Usually, patient priority maximisation and OR utilisation maximisation are the most used objectives in literature. On the contrary, the workload balance criteria, which leads to a smooth ward stay beds occupancy seems less used in literature. In this paper we propose a hierarchical multi-objective optimisation model for bed levelling and patient priority maximisation for the combined Master Surgical Scheduling and Surgical Cases Assignment problems. The aim of this work is to develop a methodology for OR planning and scheduling capable to take into account such different performance criteria.

**Keywords** Operating room planning · Elective surgery  
Multi-objective optimisation

## 1 Introduction

Operating Rooms (ORs) are one of the highest cost drivers of hospital budget and one of the highest source of income. The challenge that hospital management has to face is achieving a better and optimised use of hospital shared resources, able to reduce

---

R. Aringhieri (✉)

Dipartimento di Informatica, Università degli Studi di Torino, Turin, Italy  
e-mail: roberto.aringhieri@unito.it

P. Landa

Medical School, University of Exeter, Exeter, UK  
e-mail: P.Landa@exeter.ac.uk

S. Mancini

Dipartimento di Matematica ed Informatica, Università degli  
Studi di Cagliari, Cagliari, Italy  
e-mail: simona.mancini@unica.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_12

costs and increase profits. The adoption of decision methods as operations research can provide tools able to address the complexity of OR planning and scheduling [8, 13]. Such a complexity is given by different characteristics of the hospital organisation, where wards access to shared and limited resources such as ORs, ward beds, Intensive Care Unit (ICU) beds, Anaesthesiologists, Surgeons, Nurses. Other factors increase the complexity, such as the uncertainty of patient clinical conditions (e.g. the length of stay after the surgical intervention) and the patient surgery time in the OR session that can lead to cancellations or delays of surgery in the schedule.

OR scheduling and planning can be defined by three hierarchical decisions levels: strategic, tactical and operational that consider respectively the long, medium and short term objectives. The strategic level considers resource allocation problem, determining the number of surgeries, which staff to use for surgeries and defining the amount of the resources available. At tactical level the master surgical schedule, that is the assignment of OR blocks to surgical specialties, is defined together with the number of surgeons, the definition of ward and ICU use, and the need of equipment. Finally, at the operational decision level are defined two problems, that is (i) selecting elective patients usually from a long waiting list and assigning them to a specific OR time session (i.e., an operating room open on a specific day) over a planning horizon [10, 18], and (ii) determining the precise sequence of surgical procedures and the allocation of resources for each OR time session [15, 19]. Such problems are further challenged by the inherent stochasticity of their main parameters, such as the surgery duration, the length of stay and the arrivals of non-elective patients [1, 2, 7, 11, 12, 16, 22].

Bed availability is a topic that recently received a particular attention. Ward bed availability inside a hospital with different surgical specialties is considered in [3, 16, 22] while other studies [19, 23] consider only the use of Intensive Care Unit (ICU) or Post-Anaesthesia Care Unit (PACU) [24], or both [6, 9].

Several performance criteria have been reported to lead and to evaluate the OR planning decisions [8]. Usually, patient priority maximisation [10] and OR utilisation maximisation [14] are the most used, but also minimise delays and cancellations [16], maximise patient satisfaction [20] and minimise fixed patient costs or societal costs [23] were considered as objective function for OR planning. On the contrary, the workload balance criteria leads to a smooth—without peaks—stay bed occupancies determining a smooth workload in the ward and, by consequence, an improved quality of care provided to patients [4, 5, 21].

In this paper we propose a hierarchical multi-objective optimisation model for bed levelling and patient priority maximisation for the combined Master Surgical Scheduling and Surgical Cases Assignment problems. The aim of this work is to develop a methodology for OR planning and scheduling able to take into account such different performance criteria. The problem description and the multi-objective optimisation model are reported in Sect. 2. Preliminary computational analysis is reported and discussed in Sect. 3. Section 4 closes the paper.

## 2 Problem Statement and Mathematical Formulation

The problem addressed in this work can be formalised as follows. The goal is to simultaneously assign the OR blocks to a surgical specialty and to schedule patient surgeries in order to maximise a hierarchical objective function considering bed levelling and patients priority. More in details, the primary objective consists in maximising the number of beds occupied in a surgical specialty department, in the day in which the occupation is minimum, which represents the bottleneck of the problem. The secondary objective consists in maximising the global patients satisfaction. To each patient is assigned a score, which is computed as its priority level divided by the waiting time between the diagnosis and the surgery. The global patient satisfaction is defined as the sum of the scores related to the patients which are selected for surgery within the planning horizon.

For each patient are known the surgical specialty to which he/she is assigned, the priority level, the expected length of stay (LOS), the number of days elapsed from the diagnosis, the expected surgery duration. For each specialty is known the number of beds available on each day. Furthermore, the length of each OR block is supposed to be known. The objective is twofold. First, we try to maximise the minimum occupation of beds in a day in a department, secondly to maximise patients satisfaction, as described in the previous paragraph. A patient can be assigned to an OR block only if that block has been assigned to the surgical specialty which the patient belongs. The total expected duration of surgeries scheduled in a OR block can not exceed its length. Each scheduled patient occupies a bed in the day of his/her surgery and for a number of following days equal to his/her LOS.

Before reporting the mathematical model, we are required to introduce the following notation. Let  $I, J$  and  $K$  be respectively the sets of patients, surgical specialties and operating rooms, each indexed by  $i, j$  and  $k$ . Let  $T = \{1, \dots, N_t\}$  be the set of days in the planning horizon, indexed by  $t$ . Let  $I_j$  be the subset of patients that belong to specialty  $j, j \in J$ . For each patient  $i \in I$ , we are given the expected duration of the surgery  $p_i$ , the priority coefficient  $\pi_i$ , and the expected Length of Stay  $\mu_i$ , expressed in days. Let  $\Phi_{it}$  be the number of elapsed day between diagnosis of patient  $i$  and day  $t$ . Note that each OR block in the planning horizon is uniquely defined by the pair of indices  $(k, t)$ . We denote by  $s_{kt}$  the time capacity of the OR session  $(k, t)$ . Let  $\Lambda_{jt}$  be the number of beds available for specialty  $j$  on day  $t$ . Finally, let  $P$  and  $M$  set to  $\sum_i \pi_i$  and  $\frac{1}{P+1}$ , respectively.

Let us introduce the following decision variables: a binary variable  $X_{ikt}$  equals to 1 if patient  $i$  is assigned to block  $k$  on day  $t$ , and 0 otherwise; a binary variable  $Z_{jkt}$  equals to 1 if block  $k$  on day  $t$  has been assigned to specialty  $j$ , and 0 otherwise; a binary variable  $Y_{it}$  equals to 1 if patient  $i$  occupies a bed on day  $t$ , and 0 otherwise; a binary variable  $W_{it}$  equals to 1 if patient  $i$  surgery is scheduled on day  $t$ . Let be also  $O_1$  and  $O_2$  the primary and the secondary objective.

$$\max z = O_1 + MO_2 \quad (1a)$$

$$\text{s.t.} \quad \sum_{k \in K} \sum_{t \in T} X_{ikt} \leq 1, \quad i \in I \quad (1b)$$

$$\sum_{i \in I_j} X_{ikt} \leq |I_j| Z_{jkt}, \quad j \in J, k \in K, t \in T \quad (1c)$$

$$\sum_{j \in J} Z_{jkt} \leq 1, \quad k \in K, t \in T \quad (1d)$$

$$\sum_{i \in I} p_i X_{ikt} \leq s_{kt}, \quad k \in K, t \in T \quad (1e)$$

$$W_{it} = \sum_{k \in K} X_{ikt}, \quad i \in I, t \in T \quad (1f)$$

$$\sum_{\tau=t}^{\min(t+\mu_i; N_t)} Y_{i\tau} \geq \min(\mu_i + 1; N_t - t + 1) W_{it}, \quad t \in T \quad (1g)$$

$$\sum_{\tau=\max(t-\mu_i, 1)}^t W_{i\tau} \geq Y_{it}, \quad t \in T \quad (1h)$$

$$\sum_{i \in I_j} Y_{it} \leq \Lambda_{jt}, \quad t \in T, j \in J \quad (1i)$$

$$O_1 \leq \sum_{i \in I_j} Y_{it}, \quad t \in T, j \in J \quad (1j)$$

$$O_2 = \sum_{i \in I} \sum_{t \in T} \sum_{k \in K} \frac{\pi_i}{\Phi_{it}} X_{ikt}. \quad (1k)$$

The hierarchical objective function is reported in (1a). The role of the multiplier  $M$  is to ensure that if a solution  $S_1$  has a higher value of  $O_1$  with respect to  $S_2$  it would be preferred whichever the correspondent values of  $O_2$ . In other words, the secondary objective intervenes in the solutions comparison only when the value of  $O_1$  is exactly the same. Constraint (1b) states that only a subset of patients can be selected from the long waiting list. A patient can be assigned to an OR block only if it is assigned to the surgery specialty to which he/she belongs, as stated in constraint (1c). Constraint (1d) implies that each block must be assigned to at most one specialty. Constraint (1e) imposes that the sum of the surgery times of the patients scheduled in each OR time block  $(k, t)$  may not exceed the time block capacity  $s_{kt}$ . Constraint (1f) allows to detect whether patient  $i$  surgery is scheduled on day  $t$ . Constraints (1g) and (1h) imply that, if a patient  $i$  is scheduled on day  $t$ , he/she will occupy a bed for the following  $\mu_i$  days. Constraints (1i) limits for each specialty the number of beds occupied each day to the maximum number of available beds, given the number of beds in the department. The primary objective function (1j) concerns the maximisation of the number of beds used in the day and the specialty department with the minimal bed usage, which works as bottleneck approach. The max min bed occupation objective function tends also to implicitly fill as much as possible the

OR blocks thus avoiding under utilisation of operating rooms. The secondary objective (1k) concerns the maximisation of the patient served multiplied by the relative corresponding patient priority and divided by the waiting days from the diagnosis.

We decided to analyse also what happens if we exchange the roles of the primary and secondary objective function. This can be obtained modifying the objective function

$$\max \rho O_2 + \frac{O_1}{\lambda_{\min} + 1} \quad (2)$$

where  $\rho$  represents a multiplier constant such as, whichever value  $O_2$  takes,  $\rho O_2$  is integer. We identify with  $\lambda_{\min}$  a strict upper bound on  $O_1$ , computed as  $\lambda_{\min} = \min_{(i \in I, t \in T)} \Lambda(i, t)$ . In this way, the second term of the objective function takes always values between 0 and 1. Therefore, a solution  $S_1$  which has a higher value of  $O_2$  with respect to another solution  $S_2$ , and a value null for  $O_1$ , would results always better than  $S_2$ , whichever is the value of  $O_1$  for  $S_2$ .

### 3 Quantitative Analysis

In this section we provide a quantitative analysis in order to evaluate the behaviour of the mathematical model (1b)–(1k) varying the two objective functions (1a) and (2). To this end, we generate three sets of benchmark instances  $B_1$ ,  $B_2$  and  $B_3$  as follows: the operating time  $p_i$  case mix has been generated using the generator proposed in [17]; the elapsed time  $\Phi_{it}$  is uniformly distributed in [1,365]; the LOS  $\mu_i$  is generated accordingly to the expected surgery times, basing on the consideration that longer surgeries usually require longer stays. Five different priority levels are defined, and for each patient the priority level  $\pi_i$  is randomly generated.

The first set,  $B_1$ , is composed of 8 instances, divided in 2 groups, (1a-1b-1c-1d) and (2a-2b-2c-2d), with 200 patients uniformly distributed among 4 different specialties, each specialty has an availability of 10 beds. For each specialty we consider that some of the beds can be occupied by patients operated before the planning horizon, therefore the actual availability of beds may results lower than the capacity of the department. Priorities are different for each instance, while waiting time from the diagnosis, expected surgery duration, and LOS are constant within the same instance group, but varying among the two groups. The number of ORs is equal to 5, each one with a single block for day with a duration of 480 min. The time horizon is equal to 5 days but each OR is open only for 4 days. The day on which each OR is closed changes among ORs, such that, on each day, 4 ORs result available. The second set,  $B_2$ , contains the same instances of  $B_1$  but consider a bed capacity equal to 20 beds for each specialty. As in the previous set, actual bed availability may results lower than the capacity, due to bed occupation by patients operated before the planning horizon. Finally, the third set,  $B_3$ , is composed by 4 instances (a, b, c, d) with 400 patients and 8 specialties. Each instance is obtained merging the patients of two smaller instances in the  $B_1$  in such a way that the first instance belongs to the group 1 while the second

to the group 2. The number of ORs is 10 and their availability is defined with the same rule used for the other sets, hence 8 ORs are available on each day.

The computational results are obtained solving the model using Xpress 7.9. with a time limit of 3600 s. The computational test have been performed on a PC with a 4-core Intel i7-5500U with 2.4 GHz CPU and 16 Gb of main memory.

Table 1 reports the results for benchmark  $B_1$ . It is worth noting that the primary objective could have an impact making more challenging the solution process: actually, the solutions with patient priority maximisation (2) have a larger average gap (1.72%) and running time (3600 s) than those obtained with (1a) (0.00% and 57.89 s).

Table 2 reports the results for benchmark  $B_2$ . The results confirm the remarks for  $B_1$  even if the average gap for (2) is reduced to 0.62% and the average running time for (1a) increases up to 342.28 s.

Finally, Table 3 reports the results for benchmark  $B_3$ . We observe that for larger instances, the previous remarks are totally overturned. Although all the solutions reach the time limit, the average gap for (2) (3.26%) is smaller than the gap for (1a) (26.77%). The difference in terms of average gap could depend on whether the

**Table 1** Comparing solutions varying the objective functions (1a) and (2): benchmark  $B_1$

Instance	Objective (1a)				Objective (2)			
	$O_1$	$O_2$	Gap (%)	Time	$O_1$	$O_2$	Gap (%)	Time
I200J4B10-1a	8	5.0230	0.00	64.84	7	5.4377	0.70	3600
I200J4B10-1b	8	4.7447	0.00	51.60	7	4.8893	3.86	3600
I200J4B10-1c	8	4.4770	0.00	30.13	7	4.7799	3.29	3600
I200J4B10-1d	8	5.7638	0.00	67.44	7	6.1313	2.31	3600
I200J4B10-2a	8	5.6571	0.00	95.74	7	5.9036	1.41	3600
I200J4B10-2b	8	6.9856	0.00	64.82	5	7.0913	0.35	3600
I200J4B10-2c	8	7.5927	0.00	42.48	5	7.7670	0.14	3600
I200J4B10-2d	8	9.2648	0.00	46.03	5	9.3825	1.70	3600

**Table 2** Comparing solutions varying the objective functions (1a) and (2): benchmark  $B_2$

Instance	Objective (1a)				Objective (2)			
	$O_1$	$O_2$	Gap (%)	Time	$O_1$	$O_2$	Gap (%)	Time
I200J4B20-1a	13	5.5446	0.00	480.28	6	5.9682	0.52	3600
I200J4B20-1b	13	5.2572	0.00	444.46	6	5.4323	1.26	3600
I200J4B20-1c	13	4.6907	0.00	955.57	6	5.2938	0.50	3600
I200J4B20-1d	13	6.1827	0.00	519.12	6	6.6528	0.56	3600
I200J4B20-2a	12	6.2183	0.00	53.16	5	6.4215	0.68	3600
I200J4B20-2b	12	7.1698	0.00	186.60	0	7.7247	0.57	3600
I200J4B20-2c	12	7.8644	0.00	74.10	5	8.3281	0.41	3600
I200J4B20-2d	12	9.4352	0.00	24.97	5	9.9327	0.44	3600

**Table 3** Comparing solutions varying the objective functions (1a) and (2): benchmark  $B_3$

Instance	Objective (1a)				Objective (2)			
	$O_1$	$O_2$	Gap (%)	Time	$O_1$	$O_2$	Gap (%)	Time
I400J4B20-a	11	11.7966	21.42	3600	0	12.1370	2.79	3600
I400J4B20-b	10	12.3142	28.55	3600	0	12.5311	5.67	3600
I400J4B20-c	10	12.9874	28.55	3600	2	13.3342	2.64	3600
I400J4B20-d	10	15.9563	28.55	3600	0	16.3653	1.92	3600

instances in  $B_3$  have a larger number of patients but the same number of stay beds than those, for instance, in  $B_2$ : indeed, this situation results in an increased number of the possible patient combinations that can be operated in compliance with the operating constraints.

In terms of system performances, we can observe that taking into account only  $O_2$  the system is not able to guarantee a smooth bed occupancy. Further, considering  $O_1$  as primary objective, the solutions in terms of  $O_2$  are slightly worse than those obtained considering  $O_2$  as primary objective, while the solutions in terms of  $O_1$  are strongly better than those obtained considering  $O_2$  as primary objective, as showed in Tables 1, 2, 3. By consequence, the use of  $O_1$  as primary objective seems to lead to better global solutions.

## 4 Conclusions

In this paper we propose a hierarchical multi-objective optimisation model for bed levelling and patient priority maximisation for the combined Master Surgical Scheduling and Surgical Cases Assignment problems. The aim of this work is to develop a methodology for OR planning and scheduling capable to take into account such different performance criteria. The computational results prove the feasibility of our approach showing also a counter-intuitive result. In fact, on small instances the model where the primary objective function is the patient satisfaction results to be more difficult to solve respect to the model where the primary objective is the bed levelling, while on larger instances the model behaviour is totally overturned. This aspect will be deeply investigated with further tests. Finally, the running time required to obtain a solution suggested the need of developing ad-hoc solution algorithms.

**Acknowledgements** The authors would thank Greanne Leefink and Erwin W. Hans for providing the instance generators and the manuscript of their submitted paper [17].

## References

1. Addis, B., Carello, G., Grosso, A., Tånfani, E.: Operating room scheduling and rescheduling: a rolling horizon approach. *Flex. Serv. Manuf. J.* **28**(1–2), 206–232 (2016)
2. Aringhieri, R., Duma, D.: The optimization of a surgical clinical pathway. In: M.O. et al (ed.) *Simulation and Modeling Methodologies, Technologies and Applications. Advances in Intelligent Systems and Computing*, vol. 402, pp. 313–331. Springer (2016)
3. Aringhieri, R., Landa, P., Soriano, P., Tånfani, E., Testi, A.: A two level metaheuristic for the operating room scheduling and assignment problem. *Comput. Oper. Res.* **54**, 21–34 (2015)
4. Aringhieri, R., Landa, P., Tanfani, E.: Assigning surgery cases to operating rooms: a VNS approach for leveling ward beds occupancies. *ENDM*, pp. 173–180 (2015)
5. Beliën, J., Demeulemeester, E., Cardoen, B.: Building cyclic master surgery schedules with levelled resulting bed occupancy: a case study. *Eur. J. Oper. Res.* **176**, 1185–1204 (2007)
6. Bing-hai, Z., Meng, Y., Zhi-qiang, L.: An improved lagrangian relaxation heuristic for the scheduling problem of operating theatres. *Comput. Ind. Eng.* **101**, 490–503 (2016)
7. Bruni, M., Beraldi, P., Conforti, D.: A stochastic programming approach for operating theatre scheduling under uncertainty. *IMA J. Manag. Math.* **26**(1), 99–119 (2015)
8. Cardoen, B., Demeulemeester, E., Beliën, J.: Operating room planning and scheduling: a literature review. *Eur. J. Oper. Res.* **201**, 921–932 (2010)
9. Dellaert, N., Jeunet, J.: A variable neighborhood search algorithm for the surgery tactical planning problem. *Comput. Oper. Res.* pp. 1–10 (2017)
10. Dios, M., Molina-Pariente, J.M., Fernandez-Viagas, V., Andrade-Pineda, J.L., Framinan, J.M.: A decision support system for operating room scheduling. *Flex. Serv. Manuf. J.* **88**, 430–443 (2015)
11. Duma, D., Aringhieri, R.: An online optimization approach for the real time management of operating rooms. *Oper. Res. Health Care* **7**, 40–51 (2015)
12. Duma, D., Aringhieri, R.: The real time management of operating rooms. In: Kahraman, C., Topcu, I. (eds.) *Operations Research Applications in Health Care Management. International Series in Operations Research & Management Science*. Springer (2017) (To appear)
13. Guerriero, F., Guido, R.: Operational research in the management of the operating theatre: a survey. *Health Care Manag. Sci.* **14**, 89–114 (2011)
14. Hans, E., Wullink, G., van Houdenhoven, M., Kamezier, G.: Robust surgery loading. *Eur. J. Oper. Res.* **185**, 1038–1050 (2008)
15. Herring, W., Herrmann, J.: The single-day surgery scheduling problem: sequential decision-making and threshold-based heuristics. *OR Spectr.* **34**, 429–459 (2012)
16. Landa, P., Aringhieri, R., Soriano, P., Tånfani, E., Testi, A.: A hybrid optimization algorithm for surgeries scheduling. *Oper. Res. Health Care* **8**, 103–114 (2016)
17. Leefink, A.G., Hans, E.W.: Development of a benchmark set and instance classification system for surgery scheduling. *J. Sched.* (2017). doi:<https://doi.org/10.1007/s10951-017-0539-8>
18. Marques, I., Captivo, M., Pato, M.: An integer programming approach to elective surgery scheduling. *OR Spectr.* **34**, 407–427 (2012)
19. Meskens, N., Duvivier, D., Hanset, A.: Multi-objective operating room scheduling considering desiderata of the surgical teams. *Decis. Support Syst.* **2**(55), 650–659 (2013)
20. Min, D., Yih, Y.: Scheduling elective surgery under uncertainty and downstream capacity constraints. *Eur. J. Oper. Res.* **206**, 642–652 (2010)
21. van Oostrum, J., van Houdenhoven, M., Hurink, J., Hans, E., Wullink, G., Kazemier, G.: A master surgical scheduling approach for cyclic scheduling in operating room departments. *OR Spectr.* **30**, 355–374 (2008)
22. Siqueira, C.L., Arruda, E., Bahiense, L., Bahr, G.L., Motta, G.R.: A stochastic model for operating room planning with elective and emergency demand for surgery. *Eur. J. Oper. Res.* pp. 1–14 (2016)
23. Tånfani, E., Testi, A.: A pre-assignment heuristic algorithm for the master surgical schedule problem (mssp). *Ann. Oper. Res.* **178**(1), 105–119 (2010)
24. Vancroonenburg, W., Smet, P., Berghe, G.V.: A variable neighborhood search algorithm for the surgery tactical planning problem. *Oper. Res. Health Care* **7**, 27–39 (2015)



# Multi-Classifer Approaches for Supporting Clinical Diagnosis

Maria Carmela Groccia, Rosita Guido and Domenico Conforti

**Abstract** Clinical diagnosis processes can result in many cases very complicated. A misdiagnosis is expensive and potentially life-threatening for patients. Diagnostic problems are mainly in the scope of the classification problems. Multi-classifier approaches can improve accuracy in classification task. In this work, we propose Multi-classifier approaches based on dynamic classifier selection techniques. These approaches have been tested on datasets known in the literature and representative of important diagnostic problems. Experimental results show that a suitable pool of different classifiers increases accuracy in classification task. This suggests that the proposed approaches can improve performance of diagnostic decision support systems.

**Keywords** Multi-classifier systems · Diagnostic decision support systems  
Machine learning

## 1 Introduction

Clinical diagnosis processes involve both biomedical information gathering and clinical reasoning with the goal of determining patient's health conditions. Such information can be reported directly by the patient, acquired through physical examinations or instrumental and laboratory tests. A clinical diagnosis process can result in many cases very complex. Several factors make the process very difficult. First of all data uncertainty. Often, different diseases share many characteristics with little

---

M.C. Groccia (✉) · R. Guido · D. Conforti  
de-Health Lab, Department of Mechanical, Energy and Management  
Engineering, University of Calabria, Rende, Italy  
e-mail: mariacarmela.groccia@unical.it

R. Guido  
e-mail: rosita.guido@unical.it

D. Conforti  
e-mail: domenico.conforti@unical.it

differences and in these cases, identifying a correct diagnosis is very difficult. Clinicians need to consider more than one diagnostic hypothesis and gradually test all of them but the probability of a misdiagnosis is high. A diagnostic process can be improved with quantitative methods and technologies, properly implemented in computer systems. Under this respect, multi-classifier approaches have been proposed for supporting clinical diagnosis.

A Multi-Classifer System (MCS) is a system that uses different classifiers to perform classification task [1]. As reported in [2], a MCS is developed in three phases: generation, selection and integration. In the generation phase, a pool of classifiers, called also base classifiers, is generated. In the selection phase one or a subset of the base classifiers is selected. Finally, in the integration phase a final decision is made based on predictions of the selected base classifiers. Selection and integration phases can be facultative. Two type of classifiers selection have been proposed in the literature: static and dynamic. In a static classifiers selection, classifiers are selected at the end of the training phase and then they are always used to classify every test instance. In a dynamic classifier selection (DCS), specific classifiers are selected for each test instance, given of the assumption that each base classifier is an expert in a different local region. A local region consists of instances that are located near to a given test instance. The technique selects the most appropriate classifiers in the local region according to a competence criterion. An interesting review of dynamic selection methods is presented in [2]. The authors illustrate various dynamic selection algorithms organized according to defined competence criteria. In [3] two methods that use accuracy as competence criterion are proposed. The first method, called Dynamic Classifier Selection by Local Accuracy-Overall Local Accuracy (DCS LA OLA), evaluates the accuracy of each base classifier as the percentage of correctly labelled instances in the local region. The classifier that gets the highest accuracy is considered the most competent. The second method, called Dynamic Classifier Selection by Local Accuracy-Local Class Accuracy (DCS LA LCA), evaluates the accuracy of each base classifier on a single output class. In this case, the accuracy is defined as the percentage of instances correctly labelled in the local region and belonging to the class assigned by the classifier to the test instance. Both methods use a traditional k-Nearest Neighbours (k-NN) algorithm to define the local region, i.e., parameter  $k$  is fixed. Here we propose different MCS approaches with a dynamic classifier selection. Our contribution is twofold: (1) we define the local region of each test instance dynamically; (2) the most competent classifier is selected by a procedure based on recall and accuracy evaluated on a set of instances containing the local region.

The paper is organized as follows. Section 2 describes the proposed multi-classifier approaches. In Sect. 3, we test our approaches on publicly available diagnostic datasets, and discuss the experimental results. In Sect. 4, we show the conclusions.

## 2 Methods

In this section, we describe the proposed approaches to design a MCS based on DCS. Each approach follows the general pattern of a supervised classification process: a dataset is split in three disjunctive sets, i.e., training, validation and test set. The training set is used to train classifiers; the validation set is used to estimate the classifier's properties to recognize new instances; the test set contains the instances that have to be labelled by the classifier.

Let  $D = \{x_1, x_2, \dots, x_n\}$  be a dataset with  $n$  instances denoted by  $x_i, i = 1, 2, \dots, n$ . Each instance  $x_i = \{a_1, a_2, \dots, a_m, y_k\}$  consists of  $m$  attributes and a class label  $y_k$  from a finite set of disjoint labels  $y = \{y_1, y_2, \dots, y_k\}$ .

Each multi-classifier is composed of existing classifiers and the aim is to enhance their individual performances. A multi-classifier  $MC = \{c_1, \dots, c_h\}$  is then a pool of base classifiers  $c_j, j = 1, \dots, h$ . After the training phase of the pool of the base classifiers, we define an adaptive k-NN algorithm to compute a local region in the validation set. There are different adaptive k-NN algorithms [3, 4]. Here we propose an adaptive k-NN algorithm that selects the number of neighbours for each test instance in a dynamic way. Let  $x_i^*$  be a test instance. Our approach defines the local region of  $x_i^*$ , denoted by  $LR_{x_i^*}$ , as the set of neighbours of  $x_i^*$  in the validation set; an instance of the validation set is a neighbour only if it is within a hypersphere of radius  $R$ . The radius  $R$  is defined as

$$R = \begin{cases} (R_{max} - R_{min}) / 2 & \text{if } R_{max} > 3R_{min} \\ R_{min} & \text{otherwise} \end{cases} \quad (1)$$

where  $R_{max}$  and  $R_{min}$  is the maximum and the minimum distance, respectively, between  $x_i^*$  and all other instances in the validation set. Distances have been measured by using Euclidean norm. Depending on Eq. 1, the radius of the smallest hypersphere is  $R_{min}$ ; The biggest radius is  $\frac{R_{max}}{2}$ . The latter case occurs when  $R_{min}$  is close to zero. In this way we tried to define the local region according to the characteristics of each test instance without using trial and error. The idea is to consider local instances as close as possible to a given test instance.

Every trained classifier is then used to classify instances in the local region  $LR_{x_i^*}$ . In order to select the most competent classifier in  $LR_{x_i^*}$ , we firstly remove from the pool  $MC$  those classifiers that make mistakes on instances in  $LR_{x_i^*}$ . Then, we select the most competent classifier among the remaining classifiers. Three cases are possible after the removal phase:

1. Only one base classifier remains: it is used to classify  $x_i^*$
2. More base classifiers remain: they define the pool  $SMC \subseteq MC$ . We select the most competent classifier in  $SMC$  through criteria based on two performance indices, recall and accuracy evaluated on a given set of instances  $S_{pi}$ :

- Recall is the number of instances belonging to a class that are correctly labelled by a classifier  $c \in SMC$ . We consider the class of majority instances in  $LR_{x_i^*}$ ; let  $y_k$  be this class. Then we evaluate the recall of the classifiers in  $SMC$  on the class  $y_k$ , and the classifier with the highest recall is selected
  - Accuracy is the number of instances that are correctly labelled by a classifier  $c \in SMC$ . The classifier with the highest accuracy is selected
3. None of the base classifiers remains: we select the most competent classifier in the initial pool of base classifiers  $MC$  by the criterion based on recall or accuracy, as in case (2)

It could occur that two or more classifiers have the same best value of the chosen performance index. In order to select the best classifier, we define three different algorithms depending on the set of instances  $S_{pi}$ : Algorithm 1 uses both classifier results on training and validation sets; Algorithm 2 uses only validation results; Algorithm 3 is a hybrid of the previous ones.

### 3 Results

To show the behaviour of the proposed approaches we use three datasets from UCI Machine Learning Repository [5]: Wisconsin Diagnosis Breast Cancer (WDBC), Cleveland Heart Disease (CHD) and Dermatology dataset. Each dataset describes different features. The features in WDBC are computed from a digitized image of a fine needle aspirate of a breast mass. They describe morphological characteristics of the cell nuclei present in an image. The dataset is used to diagnose whether a breast mass is benign or malignant. Each instance in the CHD dataset describes patient's clinical parameters. The class label refers to the presence or not of heart disease. The Dermatology dataset contains both clinical and histopathological features. The dataset is used to diagnose six different types of dermatology diseases. Table 1 summarizes the main characteristics of each dataset. All datasets were normalized. We did not use feature selection.

We built several pools of base classifiers with five different classifiers: Support Vector Machines (SVM) [6], k-Nearest Neighbour (k-NN) [7], Naive Bayes (NB) [8], Decision Tree (J48) [9] and Multi-Layer Perceptron (MLP) [10]. Our multi-

**Table 1** Main characteristic of the used datasets: number of instances, number of attributes and number of classes

Dataset	Num. instances	Num. attributes	Num. classes
WDBC	569	30	2
CHD	303	13	2
Dermatology	366	34	6

**Table 2** Classification accuracy (%) of each single classifier on the datasets

Dataset	SMO	MLP	NB	J48	Ibk
WDBC	98.24	97.36	94.73	94.38	97.72
CHD	84.82	80.86	83.83	79.21	83.83
Dermatology	98.08	98.36	97.81	96.17	97.00

classifier approaches are implemented in Java whereas for each classifier we have used the related algorithm implemented in Weka [11]. A parameters tuning phase was carried out for each classifier on each dataset in order to find the best parameter values. The results of the computational experiments are related to a stratified 10 fold cross-validation, that is each dataset is split into 10 disjoint and balanced folds and the algorithms are then applied on each fold. The accuracy reported in Table 3 is the average value on 10 folds. We train and test each single classifier with the parameter values found in the tuning phase per every dataset. Table 2 shows the classifiers accuracy. In Weka, SVM is called SMO (Sequential Minimal Optimization) and k-NN is called IbK (Instance-based method with parameter k). As decision tree we use the J48 algorithm implemented in Weka.

We compare the performance of the proposed approaches with other multi-classifiers methods known in the literature, i.e., DCS LA LCA [3], DCS LA OLA [3] and Vote [12]. We used the Vote classifier of Weka and implemented in Java the versions of DCS LA LCA and DCS LA OLA reported in [3]. These two methods use a traditional k-NN algorithm to define a local region. For this reason, we constructed different local regions by varying the number of neighbours in the range 1–20, and report for each pool of base classifiers the best accuracy in this range. We tested several pools of base classifiers on each dataset. Table 3 shows the results of some of them with the best performance on the WDBC, CHD and Dermatology dataset. For each pool of base classifiers, the accuracy of Algorithm 1, Algorithm 2 and Algorithm 3 in both versions is reported, i.e. recall (Ver Rec) and accuracy (Ver Acc) as competence criterion; the best accuracy of both DCS LA LCA method and DCS LA OLA method, and the accuracy of Vote method. The best results are in bold in each row.

As reported in Table 3, in general, the proposed approaches outperform or show the same accuracy of the other tested methods. More specifically, the MCS  $MC = \{NB, SMO\}$  has the highest performance on the WDBC dataset;  $MC = \{Ibk, SMO\}$  has the highest performance on the CHD dataset; and two MCSs, i.e.,  $MC = \{SMO, MLP, NB\}$  and  $MC = \{MLP, NB, IbK\}$  have the highest performance on the Dermatology dataset. One can observe that the performance of our proposed approaches decrease when the behaviour of the classifiers in a specific pool is almost the same, since this implies that they make the same mistakes. Thus, if the number of classifiers with the same behaviour in a pool increases, the likelihood of decreasing performance is high. We can notice that, on the CHD dataset, the performance of the pool composed by three classifiers decrease with respect to the pools composed by

**Table 3** Results in term of classification accuracy (%) on the WDBC, CHD and Dermatology datasets

Base classifiers	Alg. 1		Alg. 2		Alg. 3		Best DCS LA LCA	Best DCS LA OLA	Vote
	Ver Rec	Ver Acc	Ver Rec	Ver Acc	Ver Rec	Ver Acc			
WDBC dataset									
NB-SMO	<b>98.24</b>	<b>98.24</b>	<b>98.24</b>	<b>98.24</b>	<b>98.24</b>	<b>98.24</b>	<b>98.24</b>	<b>98.24</b>	<b>98.24</b>
MLP-NB	97.36	97.36	<b>97.54</b>	<b>97.54</b>	<b>97.54</b>	<b>97.54</b>	<b>97.54</b>	97.36	96.31
NB-Ibk	97.19	<b>97.72</b>	97.01	97.01	97.01	97.01	97.19	96.31	93.5
CHD dataset									
Ibk-SMO	85.15	84.49	84.82	84.82	84.49	85.15	85.15	<b>85.48</b>	83.83
NB-Ibk	83.17	83.5	83.83	83.5	84.16	84.49	83.83	<b>85.15</b>	84.16
J48-MLP-NB	81.52	81.85	<b>83.17</b>	81.52	<b>83.17</b>	81.52	82.18	82.84	80.86
Dermatology dataset									
MLP-Ibk	98.36	98.36	<b>98.63</b>	<b>98.63</b>	<b>98.63</b>	<b>98.63</b>	98.36	98.36	96.72
SMO-MLP-NB	98.09	98.36	<b>99.18</b>	<b>99.18</b>	<b>99.18</b>	<b>99.18</b>	98.36	98.63	98.09
MLP-NB-Ibk	98.63	98.36	<b>99.18</b>	<b>99.18</b>	<b>99.18</b>	<b>99.18</b>	97.54	98.36	98.09

two classifiers. The base classifiers in the pool behave roughly in the same way and then cases of indecision are not well solved by our approaches. On the Dermatology dataset instead, the performance of the pools with three classifiers increases with respect to the performance of the pool with two base classifiers: indeed, the three base classifiers are quite different from each other in each pool and the classification accuracy improves with respect to the pool with two classifiers. In the following we analyse more in detail the results of the selected multi-classifiers on each dataset.

#### WDBC dataset

- $MC = \{NB, SMO\}$ : All methods show the same accuracy, i.e., 98.24%; this value is better or the same of the individual classifier accuracy reported in Table 2.
- $MC = \{MLP, NB\}$ : Algorithm 2 and Algorithm 3, in both versions, have the best accuracy 97.54%, which is equal to DCS LA LCA method and higher than DCS LA OLA and Vote. This MCS outperforms the individual classifiers. The

performance of Algorithm 1 is slightly lower than Algorithm 2 and 3 but the classification accuracy is not worse than the accuracy of individual classifiers.

- $MC = \{NB, Ibk\}$ : Algorithm 1 in accuracy version has the best performance 97.72% and it is not worse than the accuracy of individual classifiers.

#### CHD dataset

- $MC = \{Ibk, SMO\}$ : The best performance, for this pool of base classifiers, is achieved by Algorithm 1 in recall version and Algorithm 3 in accuracy version. Their classification accuracy 85.15% is equal to DCS LA LCA, slightly lower than DCS LA OLA and higher than Vote.
- $MC = \{NB, Ibk\}$ : All algorithms have lower performance than DCS LA OLA method. Algorithm 3 in recall versions has the best performance 85.15% which is equal to Vote, higher than DCS LA LCA and outperforms the individual classifiers.
- $MC = \{J48, MLP, NB\}$ : Algorithm 2 and 3 in recall version have the best performance 83.17%. The other algorithms have performance higher than Vote and slightly lower than DCS LA LCA and DCS LA OLA.

#### Dermatology dataset

- All MCSs outperform the individual classifiers. Algorithm 2 and 3 in both versions have always the best performance.

In general, the proposed approaches with different pools of base classifiers improve accuracy with respect to single classifier and the different methods proposed in the literature. The DCS methods define the local region using a  $k$ -nearest neighbours algorithm; the same value for the  $k$  parameter is used for all test instances and its suitable value is decided by experiments. On the contrary, the proposed approaches allow us to dynamically adapt the value of the  $k$  parameter to test instances. In particular, the  $k$  value is increased or decreased depending on characteristics of instances in the validation set. Moreover, the selection step of DCS methods uses only the accuracy of the base classifiers in the local region to select the best classifier. In our approaches instead, the most competent classifier is selected using both local accuracy of the base classifiers computed on the local region and their performance evaluated on a set of instances containing the local region.

For that concerning possible effects of training set sizes on results we observe that the WDBC dataset has 569 instances; then training and validation sets used to train and evaluate the performance of the base classifiers are sufficiently large. The CHD dataset consists of fewer instances than the WDBC dataset, so the sizes of the training and validation sets are lower. This could affect the performance of the proposed approaches. However, the achieved results are similar to the results of the other methods proposed in the literature. The Dermatology dataset has six classes not well balanced and the classes overlapping is smaller than the two previous datasets. This could explain the good reached performance with respect to the performance on WDBC and CHD datasets.

## 4 Conclusions

In this paper, we presented multi-classifier approaches for supporting clinical diagnosis. The main advantage is to exploit the diversity of every base classifier. The proposed approaches are based on a dynamic classifier selection technique. The local region is dynamically computed for each test pattern and the selection criterion is based on both misclassified instances and information about classifier's performance in a two-step process.

The experimental results suggest that a suitable combination of different classifiers can improve the performances of a diagnostic decision support system based on a single classifier. The achieved results in many cases outperform other techniques proposed in the literature.

As future work, we plan to embed these approaches in diagnostic decision support systems based on architectural organization composed by several clinical knowledge modules. In this architectural organization, each knowledge module is a MCS trained to diagnose a specific disease.

## References

1. Ranawana, R., Vasile, P.: Multi-classifier systems: review and a roadmap for developers. *Int. J. Hybrid Intell. Syst.* **3**(1), 35–61 (2006)
2. Britto, A.S., Sabourin, R., Oliveira, L.E.: Dynamic selection of classifiers—a comprehensive review. *Pattern Recognit.* **47**(11), 3665–3680 (2014)
3. Woods, K., Kegelmeyer, W.P., Bowyer, K.: Combination of multiple classifiers using local accuracy estimates. *IEEE Trans. Pattern Anal. Mach. Intell.* **19**(4), 405–410 (1997)
4. Sun, S., Huang, R.: An adaptive k-nearest neighbor algorithm. In: 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), vol. 1, IEEE (2010)
5. UCI Machine Learning Repository—<https://archive.ics.uci.edu/ml/datasets.html>
6. Vapnik, V.: *The Nature of Statistical Learning Theory*. Springer Science & Business Media (2013)
7. Kononenko, I., Kukar, M.: *Machine Learning and Data Mining: Introduction to Principles and Algorithms*. Horwood Publishing (2007)
8. Han, J., Pei, J., Kamber, M.: *Data Mining: Concepts and Techniques*. Elsevier (2011)
9. Kohavi, R., Quinlan, J.R.: Data mining tasks and methods: classification: decision-tree discovery. In: *Handbook of Data Mining and Knowledge Discovery*. Oxford University Press, Inc (2002)
10. Bishop, C.M.: *Neural Networks for Pattern Recognition*. Oxford University Press (1995)
11. Weka—<http://www.cs.waikato.ac.nz/ml/weka/>
12. Zhang, C., Ma, Y.: *Ensemble Machine Learning*, vol. 1. Springer, New York (2012)



# Offline Patient Admission Scheduling Problems

Rosita Guido, Vittorio Solina and Domenico Conforti

**Abstract** Patient admission scheduling problems consist in deciding which patient to admit and at what time. More complex problems address also bed assignment problems at the same time. The complexity of the problem motivates researchers to design suitable approaches to support bed managers in making decisions. The aim of this paper is to define an efficient model formulation for the offline elective patient admission scheduling problem, which defines admission dates and assigns patients to rooms. Due to the multiobjective nature of the problem, we suggest how to set weight values, used to penalise constraint violations. These values are tested on a set of benchmark instances. Improvements in schedule quality are presented.

**Keywords** Patient admission · Scheduling · Combinatorial optimization

## 1 Introduction

Patient admission scheduling problems (PASPs) concern with deciding which patient to admit and at what time. The patient bed assignment problem (PBAP) is a sub-task of the PASP and aims to assign patients to suitable beds by taking into account of medical constraints and patient needs. The PBAP is usually addressed as a bed capacity problem [1]. It has been formalized as an offline and combinatorial optimization problem in [2], and is an NP-hard problem as demonstrated in [3]. Ceschia and Schaerf [4] extended the PBAP of [2] by proposing an interesting dynamic version of the PASP defined as patient admission scheduling problem under uncertainty

---

R. Guido (✉) · V. Solina · D. Conforti  
de-Health Lab, Department of Mechanical, Energy and Management  
Engineering, University of Calabria, Rende, Italy  
e-mail: rosita.guido@unical.it

V. Solina  
e-mail: vittorio.solina@unical.it

D. Conforti  
e-mail: domenico.conforti@unical.it

(PASU): patient admissions can be delayed in a defined interval of days and there are some overstay risks. The goal is to define an admission date and a discharge date for elective patients by assigning them to the most suitable bed-room-wards. These authors constructed benchmark instances characterised by real-world features (e.g., uncertainty in the length of stay, registration dates, emergency patients, delayed admissions), which increase the search space of the problem.

The contribution of this study is threefold: (1) we extend our optimization models defined for PBAP in [5] to a PASP; (2) owing to the multiobjective nature of the problem, we give some tips about how to set suitable penalty costs when some constraints are violated; (3) on the basis of the new setting of penalty costs, we improve results on the benchmark instances in terms of values of constraint violations.

The paper is organized as follows. Section 2 presents the PASU and formally our optimization model, which is an improvement of the optimization models proposed in [4, 6] because we reduce at minimum the number of decision variables and take into account of constraints not considered in the previous papers. We discuss also different strategies to set penalty values. In Sect. 3 we present our computational results on a set of the benchmarks of PASU and compare and discuss them with those reported in [4]. Conclusions are drawn in Sect. 4.

## 2 Problem Statement and an Optimization Model

The PASU problem introduced in [4] is more challenging by a combinatorial point of view than the PBAP of [2]. A set of elective patients have to be admitted within a range of days to hospital and assigned to suitable rooms on the basis of their specific characteristics. This set can be selected from waiting lists by considering registration date (first come first served), patient status and such that resources are greater than demand on each day. For each patient is known an admission day that can be delayed, however before a date limit. A patient stay is given, in general, as  $L_p$  consecutive nights between admission date and discharge date. For some patients the length of stay (LOS) could be extended by some days. A transfer occurs when a patient is assigned to different rooms during her/his stay. Planned transfers are allowed, even if penalised, whenever they improve healthcare delivering to patients. Patients should be assigned to suitable rooms in correspondence with their characteristics over a planning horizon  $H$ . The reader is referred to [4, 7] for an example on PBAP and more details about PASU problem. In the following, we introduce the used notation related to departments, specialties, and patients.

A hospital has a number of departments. Each department has a set of main medical specialties and a set of auxiliary medical specialties, which correspond to high and medium levels of expertise in treating pathologies, respectively. Let  $Dep$  be the set of departments and  $S$  be the set of specialties. Some departments have age restrictions (e.g. paediatric and geriatric departments). Let  $R$ , indexed by  $r$ , be the set of rooms, and  $P$  be the set of elective patients, indexed by  $p$ , who have to be admitted

**Table 1** Room and patient attributes

Room	
$\bar{S}_r$	Set of specialties that cannot be treated in room $r$
$C_r$	Capacity, i.e., number of beds. Room capacities (e.g., single, double) define the set of room categories, denoted by $RC$
$E_r$	Set of equipment, indexed by $e$ ; $eq_{re} = 1$ if equipment $e \in E$ is in room $r \in R$ , 0 otherwise. Room capacity defines the number of equipment $e$ in a room
$gp_r \in GP$	Gender policy. There are three gender policies: the restricted gender policy (RGP), that is only male or female patients in a given room; the dependency gender policy (DGP), that is only patients with the same gender of patients already staying in a room can be assigned. We define $GP = \{1, 2, 3, 4\}$ , where 1 and 2 denote the two RGP; 3 and 4 denotes the DGP and no gender policy, respectively
$ap_{rj} \in AP$	Age policy. $AP$ is the set of age policies and $ap_{rj} = 1$ if room $r$ is subject to age policy $j \in AP$ , 0 otherwise
Patient	
$AD_p = \{a_p, \dots, a'_p\}$	Range of admission dates where $a_p$ and $a'_p$ is the first and latest possible admission date, respectively
$L_p$	Length of stay as consecutive nights
$DD_p = \{z_p, \dots, z'_p\}$	Range of discharge dates where $z_p$ is the first possible discharge date computed as $a_p + L_p$ whereas $z'_p = a'_p + L_p$ is the latest possible discharge date
$H_p = \{a_p, \dots, z'_p - 1\}$	Period during which patient has a stay of $L_p$
$sp_p \in S$	Patient specialty
Age	We set $a_{pj} = 1$ if the patient age is consistent with age policy $j \in AP$ , 0 otherwise
Health status	Defines mandatory equipment and/or preferred equipment for a patient. $me_{pe} = 1$ if equipment $e$ is mandatory, 0 otherwise
Gender	Male or female
$or_p$	Possible overstay risk in days. The LOS of patients with overstay risk could be extended by $or_p$ . For these patients $DD_p^o = DD_p \cup DD_p^{or}$ where $DD_p^{or} = \{z'_p + 1, \dots, z'_p + or_p - 1\}$

and then assigned to hospital rooms over a period  $H_p$ . The main attributes of rooms and patients are reported in Table 1.

The overall set of patients is partitioned in four subsets, depending on the last three characteristics reported in Table 1:  $P_m \subseteq P$  is the set of patients with mandatory equipment;  $P_F$  and  $P_M$  is the set of women and men, respectively;  $P_o$  is the set of patients with overstay risk.

Constraints on preferred equipment, room capacity preference, gender policies, department specialism, transfers, delayed admissions, and overcrowding risks are tackled as soft constraints and then penalised by costs. Transfers are not allowed

in overstay, thus patients remain on their last assigned room. Hard constraints are on room capacity, mandatory equipment, age policy, and patient stay as consecutive nights. The penalty values highly define schedules quality, as shown in the next section.

## 2.1 An Optimization Model for Offline Patient Admission and Scheduling Problems

Before to introduce our optimization model, we define the following decision variables, their meaning and report in Table 2 the penalty weights for the soft constraints.

$x_{prd} = 1$  if patient  $p \in P$  is assigned to room  $r \in R$  on day  $d \in H_p$ , 0 otherwise  
 $os_{prd} = 1$  if patient  $p \in P_o$  is assigned to room  $r \in R$  on day  $d \in DD_p^o$ , 0 otherwise  
 $t_{pd} = 1$  if patient  $p \in P$  is transferred on day  $d \in H_p$ , 0 otherwise  
 $ad_{pd} = 1$  if patient  $p \in P$  is admitted on day  $d \in AD_p$ , 0 otherwise  
 $b_{rd} = 1$  if both patient gender are in room  $r$  on day  $d \in H$ , 0 otherwise  
 $ov_{rd}$  overcrowding in room  $r$  on day  $d \in H$  due to possible overstay  
 $del_p \geq 0$  delayed admission (in days)

We define  $w_{pr}$  as sum of the first four penalty weights reported in Table 2.

$$\begin{aligned}
 \min \sum_{p \in P} \sum_{r \in R} \sum_{d \in H_p} w_{pr} x_{prd} &+ \sum_{r \in R_{|gpr|=3}} \sum_{d \in H} w_g b_{rd} + \sum_{p \in P} \sum_{d > d_p}^{d < z'_p} w_t t_{pd} + \sum_{p \in P} (w_{del} del_p) \\
 &+ \sum_{r \in R} \sum_{d \in H} w_o ov_{rd}
 \end{aligned} \tag{1}$$

**Table 2** Violations of the soft constraints and related penalty weights

Violation	Soft constraint	Penalty weight	Violation	Soft constraint	Penalty weight
$v_1$	Preferred equipment	$w_{pe}$	$v_5$	DGP	$w_g$
$v_2$	Room capacity preference	$w_{cr}$	$v_6$	Transfers	$w_t$
$v_3$	Department specialism	$w_{sp}$	$v_7$	Delay	$w_{del}$
$v_4$	RGP	$w_g$	$v_8$	Overcrowd risk	$w_o$

$$\sum_{d \in AD_p} ad_{pd} = 1 \quad \forall p \in P \quad (2)$$

$$\sum_{d \in H_p} x_{prd} = 0 \quad \forall p \in P_m, r \in R_{|me_{pe} > eq_{re}} \quad (3)$$

$$\sum_{d \in H_p} x_{prd} = 0 \quad \forall r \in R, p \in P_{|sp_p \in \delta_r} \quad (4)$$

$$del_p = \sum_{d \in AD_p} ad_{pd}d - a_p \quad \forall p \in P \quad (5)$$

$$\sum_{k=d}^{d+L_p-1} \sum_{r \in R} x_{prk} \geq ad_{pd}L_p \quad \forall p \in P, d \in AD_p \quad (6)$$

$$\sum_{d \in H_p} \sum_{r \in R} x_{prk} = L_p \quad \forall p \in P \quad (7)$$

$$\sum_{r \in R} x_{prd} \leq 1 \quad \forall p \in P, d \in H_p \quad (8)$$

$$C_r \geq \sum_{p \in P | d \in H_p} x_{prd} \quad \forall r \in R, d \in H \quad (9)$$

$$t_{pd} \geq x_{prd} - x_{pr(d-1)} - ad_{pd} \quad p \in P, r \in R, d \in AD_p \quad (10)$$

$$t_{pd} \geq x_{prd} - x_{pr(d-1)} \quad p \in P, r \in R, d \in ]a'_p, z'_p[ \quad (11)$$

$$\sum_{d \in ]a'_p, z'_p[} t_{pd} \leq nt \quad \forall p \in P \quad (12)$$

$$\sum_{k=0}^{or_p-1} \sum_{r \in R} os_{pr(d+L_p+k)} \geq ad_{pd}or_p \quad \forall p \in P_o, d \in AD_p \quad (13)$$

$$os_{prz_p} \leq x_{pr(z_p-1)} \quad \forall p \in P_o, r \in R \quad (14)$$

$$os_{prd} \leq x_{pr(d-1)} + os_{pr(d-1)} \quad \forall p \in P_o, r \in R, d \in ]z_p, z'_p] \quad (15)$$

$$os_{prd} \leq os_{pr(d-1)} \quad \forall p \in P_o, r \in R, d \in DD_p^{or} \quad (16)$$

$$C_r + ov_{rd} \geq \sum_{p \in P | d \in H_p} x_{prd} + \sum_{p \in P_o | d \in DD_p^o} os_{prd} \quad \forall r \in R, d \in H \quad (17)$$

$$b_{rd} \geq x_{prd} + x_{p'rd} - 1 \quad r \in R_{|gp_r=3}, d \in H, p \in P_{F|d \in H_p}, p' \in P_{M|d \in H_{p'}} \quad (18)$$

$$x_{prd} \leq a_{pj} \quad \forall p \in P, d \in H_p, j \in AP, r \in R_{|ap_{pj}=1} \quad (19)$$

$$x_{prd} \in \{0, 1\} \quad \forall p \in P, \forall r \in R, \forall d \in H_p \quad (20)$$

$$os_{prd} \in \{0, 1\} \quad \forall p \in P_o, \forall r \in R, \forall d \in DD_p^o \quad (21)$$

$$ad_{pd} \in \{0, 1\} \quad \forall p \in P, d \in AD_p \quad (22)$$

$$t_{pd} \in \{0, 1\} \quad \forall p \in P, d \in ]a_p, z'_p[ \quad (23)$$

$$b_{rd} \in \{0, 1\} \quad \forall r \in R_{|gp_r=3}, d \in H \quad (24)$$

$$ov_{rd} \geq 0 \quad \forall r \in R, d \in H \quad (25)$$

$$del_p \geq 0 \quad \forall p \in P \quad (26)$$

Objective function (1) aims to assign patients to rooms according to high quality of care and patient preferences. The first term takes into account of violations  $v_1 - v_4$  reported in Table 2. Penalty room assignments are evaluated per night by  $w_{pr}$ . The remaining terms penalise  $v_5 - v_8$ , respectively. Constraints (2) assure that each patient is admitted only once in  $AD_p$ . Constraints (3)–(4) avoid that patients with mandatory equipment are assigned to no well equipped rooms and to rooms without patient specialty, respectively. Constraints (5) define delays and Constraints (6)–(7) patient stay as consecutive  $L_p$  nights. Constraints (8) state that only one at most room is assigned to a patient on a day. Constraints (9) assure that room capacity is greater than the number of assigned patients. Constraints (10)–(11) define transfers and Constraints (12) limit their number to  $nt$ . Constraints (13)–(17) take into account of further assignments due to possible overstay of some patients. Constraints (18)–(19) are on DGP violations and age policy, respectively. Finally, Constraints (20)–(26) are integrity relations.

## 2.2 How to Set Suitable Penalty Costs

The costs in objective function (1) penalise constraint violations on preferred equipment, preference of room categories, patient specialty, gender policies, transfers, delays and overcrowding risk. These weights have to be chosen accurately such that a solver keeps the slack variables at zero, if possible. Owing the multiobjective nature of the problem, weight values are crucial for a correct solution. We provide tips regarding how to set suitable values with the following relationships among the penalty values.

$$w_g > w_t > w_{pe} = w_{sp} > w_{cr} > w_{del} > w_o \quad (27)$$

$$w_g > w_{pe} = w_{sp} > w_t > w_{cr} > w_{del} > w_o \quad (28)$$

$$w_o > w_g > w_{pe} = w_{sp} > w_t > w_{cr} > w_{del} \quad (29)$$

The above relationships have different effects on solutions: relationship (27) allows transferring a patient in a room if gender policy is not violated, whereas relationship (28) also if there are preferred features; relationship (29) highly penalises overcrowding in rooms. The default penalty values do not respect any of the relationships we have just defined. We test them on a set of instances and compare the results with those related to the default setting in the next section.

**Table 3** Violations of soft constraints and penalty values

Penalty values	Penalty values						
	$w_{pe}$	$w_{cr}$	$w_{sp}$	$w_g$	$w_t$	$w_{del}$	$w_o$
Default	20	10	20	50	100	2	1
$PV_1$	20	10	20	50	40	2	1
$PV_2$	20	10	20	50	15	2	1
$PV_3$	20	10	20	50	15	2	100

### 3 Computational Results

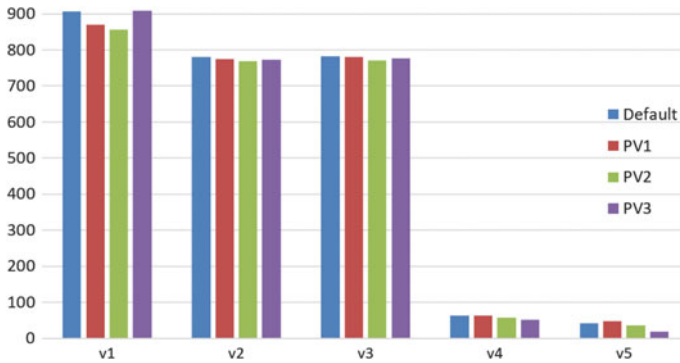
In this section we present computational results on the small short family of the benchmark instances of [4]. This family consists of 50 instances with:  $H = 14$  days,  $|Dep| = 4$ ,  $|S| = 3$ ,  $|R| = 8$ ,  $|P| = 50$ . Table 3 reports the default penalty values defined in [4], and our values defined according to relationships (27)–(29). They substantially differ for the value of  $w_t$  and  $w_o$ . We solve Model (1)–(26) by setting the maximum number of transfers per patient as  $nt = 1$ .

The schedules related to the default setting have no transfers except in Instance 19, where there is only one transfer. This is due to the high value of  $w_t$ . The schedules related to our settings have transfers: their maximum number is 4 with  $PV_1$ , and 9 with  $PV_2$ . This important aspect suggests us that we can compare these schedules with those obtained in correspondence of the default setting. Table 4 reports the mean values of the objective function and those of the constraint violations, and variations in percentage. There is a considerable improvement of the objective function values with  $PV_1$  and  $PV_2$ , excluding transfer costs, with respect to the objective function values found in correspondence to  $w_t = 100$ . The greatest improvement in decreasing order is related to constraints on preferred equipment, room capacity preference, and department specialism with the setting  $PV_1$ , and on DGP, RGP, preferred equipment, department specialism, and room capacity preference with the setting  $PV_2$ . Finally,  $PV_3$  shows a similar mean objective function value with respect to the default setting, but the lowest value for overcrowding risk. The last row of Table 4 reports the variations in percentage between the values related to  $PV_2$  and  $PV_3$  because they differ only for  $w_o$  value: although a worsening in objective function value, this last setting is the best because avoids overcrowding in rooms (i.e.,  $v_8 = 0$ ), that is possible room capacity violations. Figure 1 summarises these comparisons. The computational experiments were performed using IBM ILOG CPLEX V15.5.1; the average computational time is 150 s.

**Table 4** Mean values for the small-short instance family of PASU

Penalty values	Obj value	Constraint violations							
		$v_1$	$v_2$	$v_3$	$v_4$	$v_5$	$v_6$	$v_7$	$v_8$
Default	2605.16	906.0	779.8	783.2	63	41	2.0	26.32	3.84
$PV_1$	2592	870.4	774.4	780.8	64	47	24.8	26.40	4.04
$PV_2$	2560	855.6	769.2	770.8	57	35	41.6	26.76	4.04
$PV_3$	2605.12	908.4	772.4	776.8	51	18	49.8	28.72	0
Variations in percentage									
$\Delta\%$ (Default- $PV_1$ )	-0.51	-3.93	-0.69	-0.31	1.59	14.63	1140.00	0.30	5.21
$\Delta\%$ (Default- $PV_2$ )	-1.73	-5.56	-1.36	-1.58	-9.52	-14.63	1980.00	1.67	5.21
$\Delta\%$ ( $PV_2$ - $PV_3$ )	1.76	6.17	0.41	0.78	-10.53	-48.57	19.71	7.32	-100.00





**Fig. 1** Mean values of constraint violations  $v_1 - v_5$  with four penalty value settings

## 4 Conclusions

In this paper we defined an optimization model to manage offline patients admission scheduling under uncertainty and tested three new penalty value settings. Owing overstay uncertainty for some patients, we show that the best setting highly penalises overcrowded rooms although the other two proposed penalty value settings show lower objective function values. Future works are on testing these proposed penalty values on all the instances of PASU and solve the online problem.

## References

1. Hulshof, P., Kortbeek, N., Boucherie, R., Hans, E., Bakker, P.: Taxonomic classification of planning decisions in health care: a structured review of the state of the art in OR/MS. *Health Syst.* **1**, 129–175 (2012)
2. Demeester, P., Souffriau, W., De Causmaecker, P., Vanden Berghe, G.: A hybrid tabu search algorithm for automatically assigning patients to beds. *Artif. Intell. Med.* **48**(1), 61–70 (2010)
3. Vancroonenburg, W., Della Croce, F., Goossens, D., Spieksma, F.C.R.: The red-blue transportation problem. *Eur. J. Oper. Res.* **237**(3), 814–823 (2014)
4. Ceschia, S., Schaerf, A.: Modeling and solving the patient admission scheduling problem under uncertainty. *Artif. Intell. Med.* **3**(56), 199–205 (2011)
5. Guido, R., Groccia, M.C., Conforti, D.A.: Matheuristics for offline bed assignment problems. *Eur. J. Oper. Res.* Under Revis
6. Lusby, R.M., Schwierz, M., Range, T.M., Larsen, J.: An adaptive large neighborhood search procedure applied to the dynamic patient admission scheduling problem. *Artif. Intell. Med.* **74**, 21–31 (2016)
7. Ceschia, S., Schaerf, A.: Local search and lower bounds for the patient admission scheduling problem. *Comput. Oper. Res.* **10**(38), 1452–1463 (2011)

# Stochastic Dynamic Programming in Hospital Resource Optimization

Marco Papi, Luca Pontecorvi, Roberto Setola and Fabrizio Clemente

**Abstract** The costs associated with the healthcare system have risen dramatically in recent years. Healthcare decision-makers, especially in areas of hospital management, are rarely fortunate enough to have all necessary information made available to them at once. In this work we propose a stochastic model for the dynamics of the number of patients in a hospital department with the objective to improve the allocation of resources. The solution is based on a stochastic dynamic programming approach where the control variable is the number of admissions in the department. We use the dataset provided by one of the biggest Italian Intensive Care Units to test the application of our model. We propose also a comparison between the optimal policy of admissions and an empirical policy which describes the effective medical practice in the department. The method allows also to reduce the variability of the length of stay.

**Keywords** Stochastic dynamic programming · Healthcare resource optimization  
Hospital management

## 1 Introduction

The costs associated with the healthcare system have risen dramatically in recent years, and the increased public scrutiny to which the system has been subjected has been accompanied by increased attention from operations researchers and systems engineers. Research in this area has touched on nearly all aspects of the healthcare system, with particular emphasis being given to problems in hospital opera-

---

M. Papi · L. Pontecorvi · R. Setola (✉)  
Dipartimento di Ingegneria, Università Campus Bio-Medico, Roma, Italy  
e-mail: r.setola@unicampus.it

M. Papi  
e-mail: m.papi@unicampus.it

F. Clemente  
Istituto di Biostrutture e Bioimmagini, CNR, Napoli, Italy

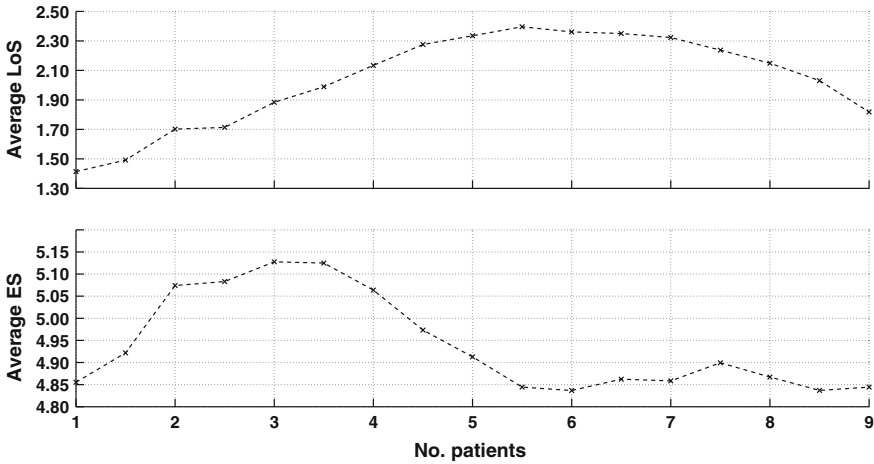
tions management [1, 2]. Healthcare decision-makers, especially in areas of hospital management, are rarely fortunate enough to have all necessary information made available to them at once. As a result, their decisions occur sequentially as information becomes available and situations around them change. These problems can be defined as sequential decision processes and solved with the use of stochastic dynamic programming (SDP). SDP represents a technique often used to solve optimization problems involving uncertainties, see [3]. Stochastic models have been successfully used in several research works to describe the allocation of resources in the healthcare system: [4–10]. In this paper we address the issue of an optimal admission strategy in the hospital department, see [8, 11, 12]. We propose a stochastic optimization model to provide an operational guideline for an Intensive Care Unit (ICU). We focus on the dynamics of the number of beds occupied and we formulate a finite horizon stochastic control problem, in discrete time, where the source of uncertainty is given by the number of patients discharged from the department. The main objective of the problem is to reduce the expected cost related to the occupancy of the ICU. A stochastic dynamic programming technique is exploited to obtain the *optimal admission policy*. Then we provide an empirical investigation based on the data of the Cardiovascular Intensive Care Unit in the *San Camillo-Forlanini* Hospital, which is one of the largest general public hospital based in Italy.

## 2 Data, Objectives and Model

The Intensive Care Unit (ICU) of *San Camillo-Forlanini* Hospital in Rome is part of the Department of Cardioscience. The data analyzed in this paper are referred to the patients coming from Cardiac Surgery Unit and they consist of daily data related to 11,770 patients (records), collected from 1999 to 2012, and the corresponding items: the Intervention Date, the Length of Stay (LoS) in the Intensive Care Unit and the EuroScore (ES). According to the literature, the LoS is considered a reliable and valid proxy for measuring the consumption of hospital resources and the ES is a synthetic risk score designed for the admission health status of a patient. As reported in Table 1, from [13], there is evidence of a close relation between the LoS, the ES and the cost of hospitalization: the higher is the ES of a patient, the lower is his health status and consequently his hospitalization will be more long and uncertain. Since hospital ward is a system characterized by a finite number of resources, if a new

**Table 1** Average cost (per day) in Cardiac Intensive Care Units in relation with Risk Score and LoS (days), from [13]

Risk Score	0	1	2	3	5	11
LoS	8.3	8.9	9.7	10.3	10.0	11.3
Cost (\$)	7,856	8,031	9,036	9,336	10,205	14,995



**Fig. 1** Average LoS and ES for different clusters of patients from San Camillo-Forlanini ICU

patient requires urgent treatment, it is reasonable to expect that less severe patients are discharged or transferred to other departments. In fact, the average LoS depends on the number of patient hospitalized in the department. We have validated this feature empirically, by considering, for each patient, the average number of patients that stay with him during his hospitalization. At each patient we have associate a proxy of the load factor<sup>1</sup> in the ward. Thus, the data have been clustered considering the average number of patients and, for each cluster, we have evaluated the average LoS. We applied a similar procedure to the evaluation of the average ES.

The results reported in Fig. 1 show, in particular, the relation between the average number of patients in the department (*x*-axes) and the average LoS (*y*-axes). There is a number of patients (around 6) that leads to the maximum LoS. The increase of the number of patients implies an increase of the LoS. However when this occupation level is achieved, the LoS starts to decrease. This feature can be explained considering that the ward is almost full and the admission demand forces physicians to discharge the patients early. Therefore, we propose a dynamic optimization approach to reduce the variability of the number of patients hospitalized in order to achieve two objectives: 1) the early allocation of resources, 2) the stability of the average LoS of patients toward values coherent with the risk profile of the hospitalized patients. To this aim we consider a dynamic stochastic model based the knowledge of the probability distribution (estimated from data) of the number of discharges from the ICU, conditional to the number of hospitalized patients.

<sup>1</sup>The ratio of actual number of hospitalized patients and the capacity of the hospital department.

**Table 2** Average and standard deviation for the number of admissions, the number of discharges and the number of patients by day of week

	No. admissions		No. discharges		No. patients	
	$\mu$	$\sigma$	$\mu$	$\sigma$	$\mu$	$\sigma$
Monday	3.11	1.53	1.77	1.33	5.09	2.07
Tuesday	2.98	1.36	2.82	1.59	5.25	2.05
Wednesday	3.10	1.40	2.91	1.44	5.44	2.12
Thursday	2.84	1.36	3.04	1.54	5.25	2.15
Friday	2.95	1.44	2.78	1.46	5.42	2.21
Saturday	0.43	0.63	1.55	1.33	4.30	1.96
Sunday	0.11	0.37	0.65	0.91	3.76	1.93

### 2.1 Conditional Distribution of the Number of Discharges

It is widely acknowledged that the number of operations, other procedure and diagnostic tests, vary from day to day. Hospital ward works 24/7 but there are peaks and troughs in activity throughout the week. Table 2 reports the mean ( $\mu$ ) and the standard deviation ( $\sigma$ ) for the number of admissions, the number of discharges and the number of patients by day of week. The values show that the average number of discharges depends strongly on the day of the week. Let  $n_k$  be the number of patients hospitalized at time (day)  $k = 1, \dots, N$ , let  $u_k$  be the number of patients discharged at time  $k$ . We define the conditional distribution of  $u_k$ , given  $n_{k-1}$ , through a Poisson model:

$$\mathbb{P}[u_k = u | n_{k-1} = n] = \frac{[\lambda_k(n)]^u e^{-\lambda_k(n)}}{u!}, \tag{1}$$

for integers  $n \geq 0$  and  $u = 0, \dots, n$ . Here  $\lambda_k(n)$  represents the average number of discharges at time  $k$ , given the number of patients ( $=n$ ) at time  $k - 1$ . Table 3 provides the estimation of  $\lambda_k$ . The increase in the number of patients corresponds to an increase in the average number of discharges (if the ward occupancy rate raises, the discharge probability raises too). Data endorse that exists a linear relationship between  $n_{k-1}$  and  $\lambda_k$ . Hence the following linear model is assumed:

$$\lambda_k(n) = m(k) \times n + q(k), \tag{2}$$

for any  $k$  and  $n$ . We also assume that the coefficients are periodic functions:  $m(k) = m(k + 7)$  and  $q(k) = q(k + 7)$ , for each  $k$ . This implies that the conditional distribution of discharges depends only on the day of week. Table 4 shows the estimated parameters for any day of week in model (2) and the AAE<sup>2</sup> between the observed average and the estimated average number of discharges.

---

<sup>2</sup>For a collection of data  $\{y_i\}_{i=1, \dots, m}$  and their estimated values  $\{\hat{y}_i\}_{i=1, \dots, m}$ , it is  $AAE = \sum_{i=1}^m \frac{|y_i - \hat{y}_i|}{m}$ .

**Table 3** Average number of discharge  $\lambda_k$  for each cluster

$n_{k-1}$	Mon	Tue	Wed	Thu	Fri	Sat	Sun
1	1.46	1.24	1.31	2.05	1.78	1.08	0.59
2	1.08	1.71	2.06	2.36	2.27	1.17	0.56
3	1.37	2.12	2.49	2.56	2.34	1.55	0.63
4	1.38	2.68	2.55	2.83	2.28	1.72	0.80
5	2.01	2.92	3.02	2.96	2.66	1.69	0.52
6	1.87	3.23	2.97	3.30	3.03	1.48	0.82
7	2.26	3.19	3.38	3.43	3.11	1.50	0.66
8	1.96	3.21	3.30	3.35	3.27	2.04	0.82
9	2.13	3.73	3.42	3.90	3.77	1.83	0.80
10	1.83	3.00	3.71	4.46	3.50	2.00	0.63

**Table 4** Estimation results for the parameters in the linear model (2)

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
$m$	0.10	0.23	0.23	0.23	0.20	0.09	0.01
$q$	1.20	1.46	1.56	1.84	1.69	1.12	0.72
AAE	0.20	0.29	0.18	0.09	0.12	0.15	0.18

### 3 Dynamics of the Number Patients

In the ICU a patient is turned away when all beds are occupied, on the other hand, the goal of the hospital is to assign beds in order to provide an adequate level of service and consumption of resources. For a given time horizon ( $N$  days), we aim to find a *policy* that specifies the action to take at each time in order to reduce the expected cost. In our model, the underlying variable is the number of patients hospitalized at time  $k$ , which follows a discrete-time dynamics:

$$n_{k+1} = n_k + e_k - u_{k+1}, \quad k = 0, \dots, N - 1, \quad (3)$$

where  $e_k$  is the number of patients admitted at time  $k$ . In order to plan beds occupancy, the decision-maker uses  $e_k$  as a control variable. In fact, physicians cannot force the discharge rate because it strictly depends on patients health status, on the other hand, they can decide the number of patients that can be admitted. Here  $u_1, u_2, \dots, u_N$  are random variables defined on the same probability space  $(\Omega, \mathcal{F}, \mathbb{P})$ , where  $\mathcal{F}$  is the  $\sigma$ -algebra of measurable events. Let  $\mathcal{F}_k$  be the  $\sigma$ -algebra generated by  $u_1, \dots, u_k$ ; we suppose that  $u_{k+1}$  is independent of  $\mathcal{F}_k$  and  $e_k$  is  $\mathcal{F}_k$ -measurable, implying that  $n_k$  is  $\mathcal{F}_k$ -measurable, for any  $k$ . In general, we assume that, for every  $k$ , there is a cost function  $g_k : \mathbb{R} \rightarrow \mathbb{R}^+$  which describes the waste of resources. The department works during the week with different load capacity, hence the total cost is assumed

to be additive in the sense that the cost incurred at time  $k$  accumulates over time. Moreover, given the presence of the source of randomness  $u_k$ , the cost is generally a random variable and cannot be meaningfully optimized. Thus, the objective function of the problem is expressed as an expected cost value:

$$\mathbb{E} \left[ \sum_{k=0}^N g_k(n_k) \right]. \quad (4)$$

The expectation is computed under the joint distribution of  $(n_1, \dots, n_N)$ . Since  $n_k$  represents the number of patients, we need to ensure that it is nonnegative and lower than the ward capacity ( $n_{\max}$ ). Thus, conditionally to  $n_{k-1} \in [0, n_{\max}]$ , the following constraints must hold: if  $n_k \leq n_{\max}$ , then  $e_{k-1} \leq n_{\max} - n_{k-1} + u_k$ , if  $n_k \geq 0$  then  $e_{k-1} \geq \max(0, u_k - n_{k-1})$ . Since  $u_k$  is random and  $e_{k-1}$  is only  $\mathcal{F}_{k-1}$ -measurable, we impose such conditions using a confidence interval. Precisely, for fixed  $\alpha, \beta \in (0, 1)$ , define

$$\bar{u}_k^\alpha \doteq \sup \left\{ \bar{u} : \mathbb{P}(u_k \leq \bar{u} | n_{k-1}) \leq \alpha \right\}, \quad \underline{u}_k^\beta \doteq \inf \left\{ \bar{u} : \mathbb{P}(u_k \geq \bar{u} | n_{k-1}) \leq \beta \right\}. \quad (5)$$

Therefore the admissible set for  $e_{k-1}$  is the set of integers in the interval  $\Theta_{k-1}(n_{k-1}) = [\underline{u}_k^\beta - n_{k-1}, n_{\max} - n_{k-1} + \bar{u}_k^\alpha]$ . Using this approach, we get the condition  $\mathbb{P}(n_k \in [0, n_{\max}] | n_{k-1} \in [0, n_{\max}]) \geq \beta - \alpha$ . We observe that  $\bar{u}_k^\alpha, \underline{u}_k^\beta$  can be computed by numerical inversion of the cumulative distribution function of  $u_k | n_{k-1}$  given by

$$\frac{\Gamma(\lfloor u + 1 \rfloor, \lambda_k(n_{k-1}))}{\lfloor u \rfloor!} \quad (6)$$

where  $\lfloor u \rfloor$  is the floor function and  $\Gamma(x, y)$  is the incomplete gamma function defined as  $\Gamma(x, y) = \int_y^\infty t^{x-1} e^{-t} dt$ .

Let  $\pi = \{\mu_0, \dots, \mu_{N-1}\}$  be the policy where  $\mu_k$  maps  $n_k$  into  $e_k = \mu_k(n_k)$  and such that  $\mu_k(n_k) \in \Theta_k(n_k)$  for all  $n_k, k = 0, \dots, N-1$ . Let  $J_\pi(n_0)$  be the expected total cost of  $\pi$ , starting at  $n_0$ , then the optimal cost function is

$$J^*(n_0) = \min_{\pi} J_\pi(n_0). \quad (7)$$

The optimal policy  $\pi^*$  satisfies  $J_{\pi^*}(n_0) = J^*(n_0)$ . According with the *principle of optimality* (see [14]),  $\pi^*$  can be constructed in piecemeal fashion, by solving a finite number of one-step optimization problems:

$$\begin{aligned} J_k(n_k) &= \min_{e_k \in \Theta_k(n_k) \cap \mathbb{N}} \mathbb{E}_k \left[ g_k(n_k) + J_{k+1} \left( g_{k+1}(n_{k+1}) \right) \right] \\ &= g_k(n_k) + e^{-\lambda_k(n_k)} \min_{e_k \in \Theta_k(n_k) \cap \mathbb{N}} \sum_u \frac{1}{u!} J_{k+1}(n_k + e_k - u) [\lambda_k(n_k)]^u \end{aligned} \quad (8)$$

for  $k = N - 1, \dots, 0$ , starting from the terminal cost  $J_N(\cdot) \equiv 0$ . Here  $\mathbb{E}_k$  is the expected value conditional to  $\mathcal{F}_k$ .  $J_0(n_0)$ , generated at the last step, is equal to the optimal cost  $J^*(n_0)$  and the collection of the policies that minimize the functionals in (8),  $\pi^* = \{\mu_0^*(n_k), \dots, \mu_{N-1}^*(n_k)\}$ , represents the optimal policy for the whole minimization problem. It is easy to show that, in presence of convex cost functions  $g_k$ ,  $k = 0, \dots, N$ , the value functions  $J_k$  is also convex for all  $k = 0, \dots, N$ . Therefore the program is convex and general methods from convex optimization can be used in this case.

### 3.1 Empirical Results

Our empirical investigation is based on the choice of the cost function  $g_k(n) = (n - \bar{n}_k)^2$ , where  $\bar{n}_k$  is a target number of hospitalizations at time  $k$ . Using this cost function we also pursue the reduction of the variability for the number of patients according with the observations made in Sect. 2. We evaluate the stochastic dynamic algorithm (*SDP*) in comparison with an empirical policy (*LP*), considered as a proxy of the practice commonly applied in the department, that is based on the average number of discharges:

$$\tilde{e}_k = \left[ \bar{n}_k - n_k + \mathbb{E}[u_{k+1}|n_k] \right]^+, \quad (9)$$

We compare *SDP* and *LP* policies for different values of the desired weekly average number of patients ( $\bar{n}$ ). Here the daily target number of patients is described as follows:

$$\bar{n}_k = \begin{cases} \lfloor \bar{n} \times w_k \rfloor & \text{if } \{ \bar{n} \times w_k \} \leq 0.5, \\ \lfloor \bar{n} \times w_k \rfloor + 1 & \text{if } \{ \bar{n} \times w_k \} > 0.5, \end{cases} \quad (10)$$

where  $\{\cdot\}$  denotes the fractional part of a number and  $w = [1.04 \quad 1.06 \quad 1.10 \quad 1.06 \quad 1.09 \quad 0.87 \quad 0.77]$  is computed according to an historical analysis of the flow of patients in the ward. Setting  $\alpha = 0.03$  and  $\beta = 0.97$  for the constraint on the control variables, we have considered  $10^4$  simulated scenarios over a time horizon of 28 days (4 weeks) in which the number of discharges are simulated according to the proposed model. For each scenario, the relative average absolute distance between the required number of patients and the hospitalized number is evaluated:  $\epsilon = [1/(28 \cdot \bar{n})] \cdot \sum_{k=1}^{28} |n_k - \bar{n}_k|$ . The numerical results (in percentage) are reported in Table 5 together with the optimal SDP policy for  $\bar{n} = 4$ . Given the periodicity of the probability distribution of  $u_k$ , we have reported only the optimal control for a week. From the estimated error, we argue that there is evidence of a better performance of the SDP strategy with respect to the LP policy, the difference being always greater than 10%. We conclude that the application of the SDP policy could lead to a substantial improvement in the planning of resources in the ICU.



**Table 5** Relative average absolute distance between the target and the hospitalized number of patients and the SDP policy obtained considering  $\bar{n} = 4$

	$\bar{n}$							
	4	5	6	7	8	9	10	11
SDP	27.54	23.71	18.63	19.43	16.96	16.19	14.85	14.35
LP	36.04	32.21	29.78	30.98	27.54	26.22	25.5	26.07

$n_{k-1}$	Mon	Tue	Wed	Thu	Fri	Sat	Sun
1	4	4	5	5	4	4	3
2	3	3	4	4	3	3	2
3	2	2	3	3	2	2	1
4	1	2	2	2	2	1	0
5	0	1	1	1	1	0	0
6	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0

## 4 Conclusions

Hospital management takes place in an increasingly competitive environment and it is therefore essential to focus on delivering high quality care to patients. The aim of the hospital bed management is to allocate resources for patients while taking into account capacity constraints. The study was conducted on an Italian Cardiology Intensive Care Unit. In this work we propose a stochastic dynamic programming (SDP) approach to control the number of admissions where the uncertainty is driven by the number of discharges. The main objective consists in the minimization of an expected cost function over a finite horizon. A comparison with a proxy of the medical practice used in the ICU is also conducted. The numerical results show that through the implementation of the SDP optimal policy, hospital managers could balance the cost of empty beds against the cost of turning patients away, thus facilitating a good choice of beds provision, in order to have a low cost and high access to service. This method allows to keep under control beds occupancy rate, the length of stay (LoS) in the hospital department.

**Acknowledgements** This research is supported in part by San Camillo-Forlanini Hospital in Rome.

## References

1. Hans, E.W., Van Houdenhoven, M., Hulshof, P.J.: A framework for healthcare planning and control. In: *Handbook of Healthcare System Scheduling*, vol. 168, pp. 303–320 (2012)
2. Harper, P., Shahani, A.: Modelling for the planning and management of bed capacities in hospital. *J. Oper. Res. Soc.* **53**, 11–18 (2002)
3. Herring, W.L.: Prioritizing patients: stochastic dynamic programming for surgery scheduling and mass casualty incident triage. *Doctoral Dissertations* (2011)
4. Castaing, J., Cohn, A., Denton, B., Weizer, A.: A stochastic programming approach to reduce patient wait times and overtime in an outpatient infusion center. *IIE Trans. Healthc. Syst. Eng.* **6**(3), 111–125 (2015)
5. Erdelyi, A., Topaloglu, H.: Approximate dynamic programming for dynamic capacity allocation with multiple priority levels. *IIE Trans.* **43**(2), 129–142 (2010)
6. Ozen, A.: Stochastic models for capacity planning in healthcare delivery: case studies in an outpatient, inpatient and surgical setting. *Doctoral Dissertations 2014-current*. Paper 125 (2014)
7. Punnakitikashem, P., Rosenberger, J.M., Behan, D.B.: Stochastic programming for nurse assignment. *Comput. Optim. Appl.* **40**(3), 321–349 (2008)
8. Dong, M., Li, J., Zhao, W.: Admissions optimization and premature discharge decisions in intensive care units. *Int. J. Prod. Res.* **53**, 7329–7342 (2015)
9. Gilleskie, D.B.: A dynamic stochastic model of medical care use and work absence. *Econometrica* **66**, 1–45 (1998)
10. Xiao, G., van Jaarsveld, W., Dong, M., van de Klundert, J.: Stochastic programming analysis and solutions to schedule overcrowded operating rooms in China. *Comput. Oper. Res.* **74**, 78–91 (2016)
11. Gorunescu, F., McClean, S.I., Millard, P.H.: A queueing model for bed-occupancy management and planning of hospital. *J. Oper. Res. Soc.* **53**, 19–24 (2002)
12. Green, L.V.: Capacity planning and management in hospital. *Oper. Res. Health Care* **70**, 15–44 (2004)
13. Kurki, T.S., Hakkinen, U., Lauharanta, J., Ramo, J., Leijala, M.: Euroscore predicts health-related quality of life after coronary artery bypass grafting. *Interact. Cardiovasc. Thorac. Surg.* **7**, 564–568 (2008)
14. Bertsekas, D.: *Dynamic Programming and Optimal Control*, vol. 2, 4th edn. Athena Scientific (2011)

**Part IV**  
**Heuristics and Metaheuristics**

# Column Generation Embedding Carousel Greedy for the Maximum Network Lifetime Problem with Interference Constraints

Francesco Carrabs, Carmine Cerrone, Ciriaco D'Ambrosio  
and Andrea Raiconi

**Abstract** We aim to maximize the operational time of a network of sensors, which are used to monitor a predefined set of target locations. The classical approach proposed in the literature consists in individuating subsets of sensors (*covers*) that can individually monitor the targets, and in assigning appropriate activation times to each cover. Indeed, since sensors may belong to multiple covers, it is important to make sure that their overall battery capacities are not violated. We consider additional constraints that prohibit certain sensors to appear in the same cover, since they would interfere with each other. We propose a Column Generation approach, in which the pricing subproblem is solved either exactly or heuristically by means of a recently introduced technique to enhance basic greedy algorithms, known as Carousel Greedy. Our experiments show the effectiveness of this approach.

**Keywords** Column generation · Carousel greedy · Maximum lifetime problem

## 1 Introduction

The Maximum Lifetime Problem (MLP) and its variants have been the focus of many studies in the last years. Given a geographical region in which some important target locations (or simply *targets*) have been individuated, the aim is to use a net-

---

F. Carrabs · C. D'Ambrosio · A. Raiconi (✉)  
Department of Mathematics, University of Salerno,  
Via Giovanni Paolo II 132, 84084 Fisciano, SA, Italy  
e-mail: araiconi@unisa.it

F. Carrabs  
e-mail: fcarrabs@unisa.it

C. D'Ambrosio  
e-mail: cdambrosio@unisa.it

C. Cerrone  
Department of Biosciences and Territory, University of Molise,  
Contrada Fonte Lappone, 86090 Pesche, IS, Italy  
e-mail: carmine.cerrone@unimol.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_16

work of sensors for as long as possible to keep these locations under observation. We assume the sensors to be located in advance, as opposed to other works focusing on their placement (see [20]). The *cover* is a key concept; with this term, we refer to a subset of sensors that is independently able to monitor all targets. A common approach for facing MLP problems is the following: find a collection of covers, and activate them one at a time. By activating a cover, we mean that all of its sensors are switched to their sensing mode, while all other sensors are kept in an idle, energy-saving state. Clearly, if a sensor belongs to more than a cover, the overall activation times of these covers cannot exceed the maximum activation time imposed by the sensor battery. Among the first works dealing with the MLP problem, we recall [3]. In this work, it is first shown that allowing non-disjoint covers can bring noticeable improvements in terms of network lifetime. The authors also demonstrate that the problem is NP-complete, and present LP-based and greedy heuristics. A resolution approach to solve MLP based on Column Generation was presented in [15]. Several MLP variants have been proposed and studied. In some works, sensors are allowed to enlarge their sensing radii at the expense of additional energy consumption [13, 14, 19]. Other lines of research consider the case in which it is allowed to leave some targets uncovered [7, 16, 21], sensors belong to different types [2, 5], or connectivity issues are taken into account [1, 6, 8, 10, 18]. Fewer research efforts in this context considered interference issues. Concurrent transmission of sensors that are too close may cause data collision, which in turn is responsible for data loss and additional energy expense; see for instance [17]. In [9], the authors present the Maximum Lifetime Problem with Interference Constraints (MLIC). In this problem, a collection of pairs of *conflicting* sensors are considered. For each of these pairs, at most one sensor can belong to any given cover. The authors present a Column Generation algorithm, whose pricing subproblem is solved either exactly or heuristically by means of a greedy heuristic. In this work, we modify the algorithm presented in [9] by facing the heuristic resolution of the pricing subproblem (called simply subproblem from now on) through Carousel Greedy, a novel paradigm for enhancing greedy heuristics, originally proposed in [12]. The next sections illustrate the general Column Generation approach for MLIC (Sect. 2), the proposed Carousel Greedy procedure (Sect. 3), the results of our tests (Sect. 4) and our final remarks (Sect. 5).

## 2 Column Generation Approach

Let  $S = \{s_1, \dots, s_m\}$  be the set of the sensors, and  $T = \{t_1, \dots, t_m\}$  be the target points. A cover  $C_k$  for the MLIC problem is a subset of  $S$ , such that each target  $t_j \in T$  is within the sensing area of at least one sensor  $s_i \in C_k$  (it is *monitored*, or *covered* by it), and such that every couple of sensors  $(s_i, s_j) \in C_k \times C_k$  is not a conflicting pair. As mentioned in the introduction, the objective is to maximize the network lifetime by assigning appropriate activation times to covers. The overall number of covers can be exponential in size; hence, in [9], a Column Generation (ColGen) approach was proposed in order to implicitly discard most of them. Let  $\mathcal{C} = \{C_1, \dots, C_z\}$  be

a set composed by some feasible covers for MLIC. In [9], the master problem of the ColGen approach for the MLIC problem is formulated as follows:

$$[\mathbf{MP}] \max \sum_{C_k \in \mathcal{C}} w_k \quad (1)$$

$$\sum_{C_k \in \mathcal{C}: s_i \in C_k} w_k \leq \tau_i \quad \forall s_i \in S \quad (2)$$

$$w_k \geq 0 \quad \forall C_k \in \mathcal{C} \quad (3)$$

Variables  $w_k$  represent for how long each cover  $C_k \in \mathcal{C}$  is kept in active state in the solution, while parameter  $\tau_i \forall s_i \in S$  represents the maximum amount of activation time available for the sensor, given its battery capacity. The column of each variable  $w_k$  in the coefficient matrix contains a 1 in the  $i$ -th position if  $s_i \in C_k$ , 0 otherwise. It is then clear that the optimal **[MP]** solution represents the maximum lifetime that can be obtained by considering the subset of covers contained in  $\mathcal{C}$ , while respecting the battery duration constraints. In order to evaluate whether this is also the optimal solution for the whole problem, we need to solve a subproblem to identify the nonbasic variable with maximum reduced cost. The subproblem formulation, also presented in [9], is the following:

$$[\mathbf{SP}] \max 1 - \sum_{s_i \in S} \pi_i y_i \quad (4)$$

$$\sum_{s_i \in S: \delta_{ij}=1} y_i \geq 1 \quad \forall t_j \in T \quad (5)$$

$$y_i + y_j \leq 1 \quad \forall (s_i, s_j) \in S \times S : \gamma_{ij} = 1, i < j \quad (6)$$

$$y_i \in \{0, 1\} \quad \forall s_i \in S \quad (7)$$

The subproblem needs to build the column associated with a cover, therefore each variable  $y_i \forall s_i \in S$  will be equal to 1 if the sensor is chosen to belong to it, and 0 otherwise. Parameters  $\pi_i$ , weighting the  $y_i$  variables in the objective function, represent the dual prices associated with the sensors after solving **[MP]**. The coefficient of each new entering variable in the objective function of the master problem will always be 1, hence the constant value in (4). Each parameter  $\delta_{ij}$  is equal to 1 if sensor  $s_i \in S$  can monitor target  $t_j \in T$  and 0 otherwise. Finally, each parameter  $\gamma_{ij} \forall (i, j) \in S \times S$  is equal to 1 if  $(s_i, s_j)$  is a conflicting pair and 0 otherwise. It follows that constraints (5) and (6) make sure that the chosen sensors define a cover, while the objective function maximizes its reduced cost. In particular, if  $1 - \sum_{s_i \in C} \pi_i < 0$  for the newly built cover  $C$ , then the solution found by **[MP]** was optimal for the MLIC problem. Otherwise,  $C$  could potentially be used to find a better solution (it is defined to be an *attractive* cover); it is added to the set  $\mathcal{C}$  and the ColGen algorithm iterates. The main drawback of this approach is that the subproblem is NP-Hard itself. Hence, in [9], a greedy heuristic for the subproblem is presented. In this work we enhance this heuristic by transforming it into a Carousel Greedy, which is presented in next section.

### 3 Carousel Greedy Approach for the Subproblem

The Carousel Greedy is a generalized method to enhance greedy algorithms, originally proposed in [12]. The aim is to obtain a procedure which is almost as fast and simple as the greedy procedure on which it is based, while achieving accuracy levels similar to those of a metaheuristic. The authors show the effectiveness of their proposal for several classical combinatorial optimization problems. The main observation underlying Carousel Greedy is that during the execution of a constructive heuristic, the later decisions are likely to be more informed and valid than the earlier ones. Given this observation, a Carousel Greedy procedure increases the solution space visited by a basic greedy, operating in three main steps:

- In the first step a partial (unfeasible) solution is built. The first step ends when the partial solution size reaches a given percentage of the size of a complete (feasible) solution.
- In the second step, the partial solution is modified by iteratively removing from it the oldest choices and making new ones. The second step ends after a pre-defined number of iterations.
- In the final step, the partial solution is completed to produce a feasible solution.

Our proposed Carousel-SP algorithm enhances Greedy-SP, a constructive heuristic presented in [9] to solve the MLIC subproblem (in both names, SP stands for subproblem). Carousel-SP works as follows:

- The first step starts from an empty set  $C$ , and iteratively adds sensors to it. The sensors are chosen from a set of candidates  $S_c$ , representing sensors that are not in  $C$  and are not in conflict with any of its elements. Therefore,  $S_c$  is initialized with  $S$ . At each iteration, the algorithm uses a greedy criterion to select the next sensor to be added to  $C$ . In more detail, it selects the sensor  $s_i \in S_c$  that minimizes the quantity  $\omega_i = \frac{\pi_i}{|T_i|}$ , where  $\pi_i$  is the dual price of the sensor and  $|T_i|$  the amount of additional targets that would be monitored by  $C$  by adding  $s_i$  to it. The greedy criterion is designed to favor sensors with low dual prices that cover many targets. After adding  $s_i$  to  $C$ , the elements of  $S_c$  that form a conflicting pair with  $s_i$  (as well as  $s_i$  itself) are removed from  $S_c$ . This step ends as soon as the number of uncovered targets is equal to or lower than  $\beta|T|$ , with  $\beta \in [0, 1]$ . If  $S_c$  becomes empty first, Carousel-SP ends in failure.
- In each iteration of the second step, the sensor in  $C$  corresponding to the oldest choice is removed from it, and is replaced with a new one. After the removal of a sensor  $s_i$  from  $C$ , the set  $S_c$  is updated to include  $s_i$  and any sensor that was removed because it was in conflict with  $s_i$ . Since some new targets may become uncovered after the  $s_i$  removal, the  $\omega_i$  values are updated as well. Each new sensor added to  $C$  is selected according to the same greedy criterion used for the first step. Eventually, after one or more iterations of the second step,  $C$  may become a feasible solution (that is, a cover); if this is the case, the feasible solution with the lowest objective function value ( $\sum_{s_i \in C} \pi_i$ ) is stored as incumbent solution  $C'$ . Furthermore, after having found a cover, two sensors instead than one are removed from

$C$  in the following iteration, to avoid cycling on the same solution. The second step is iterated  $\alpha h$  times, with  $\alpha \geq 1$  and  $h = |C|$  at the end of the first step. After the last iteration, if an incumbent solution  $C'$  exists, it is returned and Carousel-SP ends its execution.

- The third step operates similarly to the first one, with two differences; it starts from the  $C$  set returned by the second step, and it iterates until all targets are covered. As soon as  $C$  becomes a feasible solution, it is returned, and Carousel-SP ends. If  $S_c$  becomes empty first, Carousel-SP ends in failure.

We now discuss how Carousel-SP is integrated within the ColGen framework. In each iteration, after solving [MP], we attempt to solve the subproblem using Carousel-SP. If it returns an attractive cover  $C$ , it is added to  $\mathcal{C}$  and the current ColGen iteration ends. Otherwise, if Carousel-SP fails or if it returns an unattractive cover, we solve the subproblem exactly using [SP]. Again, if an attractive cover is found, it is added to  $\mathcal{C}$  and a new ColGen iteration begins. Otherwise, the optimality for the MLIC problem of the last solution found by [MP] has been proven.

## 4 Computational Results

In this section we compare our approach (CG+C), that embeds Carousel-SP, with the ColGen algorithm (CG+G) proposed in [9], which uses Greedy-SP. Both CG+G and CG+C were coded in C++. All tests were run on a Linux machine with an Intel Xeon E5-2650 CPU running at 2.30 GHz and 128 GB of RAM. For both the approaches, the Concert library of IBM ILOG CPLEX 12.6.1 was used to solve the mathematical formulations. Tests were run in single thread mode, with a time limit of 1 hour for each test. We considered the same set of instances used in [9], with a number of sensors varying in the set {300, 400, 500, 750, 1000, 1250}, and either 15 or 30 targets. All sensors have a battery duration normalized to 1 time unit. Sensors and targets are disposed in a square area with size  $500 \times 500$ . Each sensor has a *sensing* ( $RS$ ) and a *conflict* ( $RC$ ) range.  $RS$  is equal to either 100 or 125; targets with an euclidean distance within this value from a sensor are covered by it.  $RC$  is equal to 175; two sensors within this distance from each other form a conflicting pair. There are 4 different instances for each combination of parameters, for a total of 96 instances. Note that the computational test carried out in [9] also considered the case  $RC = 125$ . However, the authors have shown that these instances are usually very easily solved; in particular, for  $RS = 125$ , the number of subproblems solved to optimality is often equal to 1. In this context, it is pointless to apply Carousel-SP, since it is more expensive than Greedy-SP and there are not margins to speed up the convergence of the ColGen algorithm. Therefore, we report in the following the computational results only for  $RC = 175$ . Regarding the Carousel-SP parameters, after a preliminary test, the values  $\alpha = 3$  and  $\beta = 0.2$  were chosen.



**Table 1** Computational results on the small instances with  $RC = 175$ 

	S	T	CG+G			CG+C			
			LF	Time	SubInv	LF	Time	SubInv	Gap (%)
RS = 100	300	15	13.75	4.57	75.50	13.75	4.66	74.25	
	300	30	9.16	9.35	72.25	9.16	10.10	74.25	
	400	15	18.25	12.11	84.75	18.25	12.71	89.25	
	400	30	14.25	34.84	127.25	14.25	33.32	123.50	4.36
	500	15	25.50	42.36	177.75	25.50	39.20	164.25	7.46
	500	30	19.00	118.05	204.25	19.00	121.56	200.75	-2.97
RS = 125	300	15	23.25	3.89	54.00	23.25	3.29	39.25	
	300	30	18.25	7.17	83.25	18.25	7.25	76.25	
	400	15	32.50	5.59	32.75	32.50	6.83	29.50	-22.18
	400	30	26.75	30.48	150.50	26.75	28.95	138.75	5.02
	500	15	41.25	35.52	122.00	41.25	29.23	87.50	17.71
	500	30	38.00	85.67	228.25	38.00	88.24	226.25	-3.00

The results of the comparison between the two algorithms on the smaller instances ( $|S| \leq 500$ ) are reported in Table 1, containing 12 rows split in 2 groups of 6 rows each, associated with  $RS = 100$  and  $RS = 125$ , respectively. The first two columns show the number of sensors ( $|S|$ ) and targets ( $|T|$ ) in the scenarios. The next 6 columns report, for each algorithm, the lifetime rounded to 2 decimal digits ( $LF$ ), the computational time in seconds ( $Time$ ) and the number of iterations with heuristic fails, in which therefore the subproblem has to be solved to optimality ( $SubInv$ ), respectively. Each entry in these columns is an average value for the 4 instances of a given scenario. Finally, the last column shows the percentage gap ( $Gap$ ) between the computational times, evaluated as  $\frac{Time(CG+G) - Time(CG+C)}{Time(CG+G)}$ . When the gap is lower than 1 second, we consider it negligible and therefore we do not report its percentage value. All the small scenarios are solved to optimality by both algorithms, and hence we only focus on performances. On 5 scenarios, the time gap is negligible. On the remaining scenarios, CG+C is faster than CG+G 4 times, with a percentage gap that ranges from 4.36 to 17.71%, and is slower 3 times, with a percentage gap ranging from 2.97 to 22.18%. Regardless of percentage gaps, the performances of the two algorithms are very close for all these scenarios, since the time gap is always lower than 7 s. The results obtained on the larger scenarios are more interesting. These results are reported in Table 2. As expected, the computational times of CG+C and CG+G are higher than the ones required to solve small scenarios, and there is a scenario (marked with a “\*” symbol) that is not solved within the time limit by CG+G ( $|S| = 1250$ ,  $|T| = 15$  and  $RS = 100$ ). The scenario is, instead, solved to optimality by CG+C in around half an hour. CG+C results only once slower than CG+G, on the scenario with  $|S| = 1000$ ,  $|T| = 30$  and  $RS = 125$ , with a percentage gap equal to 0.51%, corresponding to about 7 s. On all the remaining scenarios CG+C is faster

**Table 2** Computational results on the large instances with  $RC = 175$

	S	T	CG+G			CG+C			
			LF	Time	SubInv	LF	Time	SubInv	Gap (%)
RS = 100	750	15	68.50	671.88	454.50	68.50	582.77	426.25	13.26
	750	30	43.00	719.42	421.25	43.00	696.33	419.50	3.21
	1000	15	60.25	962.01	257.25	60.25	487.45	120.00	49.33
	1000	30	49.00	1704.59	499.50	49.00	1610.24	481.25	5.54
	1250	15	85.91*	2274.69	453.50	86.75	2014.62	425.25	11.43
	1250	30	52.75	2365.43	406.50	52.75	2145.40	389.75	9.30
RS=125	750	15	72.25	117.82	79.00	72.25	98.99	55.75	15.98
	750	30	55.50	419.09	237.75	55.50	391.86	232.25	6.50
	1000	15	96.75	796.88	239.75	96.75	660.31	181.25	17.14
	1000	30	79.00	1360.21	406.50	79.00	1367.19	403.50	-0.51
	1250	15	127.75	843.91	147.50	127.75	562.74	95.25	33.32
	1250	30	68.25	2035.12	357.00	68.25	1707.32	321.75	16.11

than CG+G, with a percentage gap that ranges from 3.21 to 49.33% and a time gap up to about 500 seconds. In particular, for 7 out of 12 scenarios, CG+C results at least 10% faster than CG+G. The higher effectiveness of Carousel-SP is testified by the smaller SubInv values of CG+C with respect to CG+G; this affects significantly the related computational times. For instance, on the scenario with  $|S| = 1000$ ,  $|T| = 15$  and  $RS = 100$ , the SubInv value and the computational time for CG+G are about twice greater than the values for CG+C. Similar observations can be done for the other scenarios.

## 5 Conclusions

We proposed a column generation algorithm to solve the Maximum Lifetime Problem with Interference Constraints. We improve a previous algorithm by introducing a new method to solve heuristically the subproblem, based on the Carousel Greedy paradigm. Computational tests show the effectiveness of our proposal, in particular for larger test instances. Further research will be focused on improving the Carousel Greedy procedure through hybridization with metaheuristic approaches, such as Tabu Search and Genetic Algorithm [4, 11].

## References

1. Alfieri, A., Bianco, A., Brandimarte, P., Chiasserini, C.F.: Maximizing system lifetime in wireless sensor networks. *Eur. J. Oper. Res.* **181**(1), 390–402 (2007)
2. Awada, W., Cardei, M.: Energy-efficient data gathering in heterogeneous wireless sensor networks. In: Proceedings of the IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, pp. 53–60 (2006)
3. Cardei, M., Thai, M.T., Li, Y., Wu, W.: Energy-efficient target coverage in wireless sensor networks. In: Proceedings of the 24th Conference of the IEEE Communications Society, vol. 3, pp. 1976–1984 (2005)
4. Carrabs, F., Cerrone, C., Cerulli, R.: A tabu search approach for the circle packing problem. In: 2014 17th International Conference on Network-Based Information Systems (NBIS), pp. 165–171 (2014)
5. Carrabs, F., Cerulli, R., D’Ambrosio, C., Gentili, M., Raiconi, A.: Maximizing lifetime in wireless sensor networks with multiple sensor families. *Comput. Oper. Res.* **60**, 121–137 (2015)
6. Carrabs, F., Cerulli, R., D’Ambrosio, C., Raiconi, A.: An exact algorithm to extend lifetime through roles allocation in sensor networks with connectivity constraints. *Optim. Lett.* (to appear). <https://doi.org/10.1007/s11590-016-1072-y>
7. Carrabs, F., Cerulli, R., D’Ambrosio, C., Raiconi, A.: A hybrid exact approach for maximizing lifetime in sensor networks with complete and partial coverage constraints. *J. Netw. Comput. Appl.* **58**, 12–22 (2015)
8. Carrabs, F., Cerulli, R., D’Ambrosio, C., Raiconi, A.: Extending lifetime through partial coverage and roles allocation in connectivity-constrained sensor networks. *IFAC-PapersOnline* **49**(12), 973–978 (2016)
9. Carrabs, F., Cerulli, R., D’Ambrosio, C., Raiconi, A.: Prolonging lifetime in wireless sensor networks with interference constraints. In: Au, M., Castiglione, A., Choo, K.K., Palmieri, F., Li, K.C. (eds.) *Green, Pervasive, and Cloud Computing*. GPC 2017. Lecture Notes in Computer Science, vol. 10232, pp. 285–297. Springer, Cham (2017)
10. Castaño, F., Rossi, A., Sevaux, M., Velasco, N.: A column generation approach to extend lifetime in wireless sensor networks with coverage and connectivity constraints. *Comput. Oper. Res.* **52**(B), 220–230 (2014)
11. Cerrone, C., Cerulli, R., Gaudio, M.: Omega one multi ethnic genetic approach. *Optim. Lett.* **10**(2), 309–324 (2016)
12. Cerrone, C., Cerulli, R., Golden, B.: Carousel greedy: a generalized greedy algorithm with applications in optimization. *Comput. Oper. Res.* **85**, 97–112 (2017)
13. Cerulli, R., De Donato, R., Raiconi, A.: Exact and heuristic methods to maximize network lifetime in wireless sensor networks with adjustable sensing ranges. *Eur. J. Oper. Res.* **220**(1), 58–66 (2012)
14. Cerulli, R., Gentili, M., Raiconi, A.: Maximizing lifetime and handling reliability in wireless sensor networks. *Networks* **64**(4), 321–338 (2014)
15. Deschinkel, K.: A column generation based heuristic for maximum lifetime coverage in wireless sensor networks. In: 5th International Conference on Sensor Technologies and Applications SENSORCOMM 11, vol. 4, pp. 209–214 (2011)
16. Gentili, M., Raiconi, A.:  $\alpha$ -coverage to extend network lifetime on wireless sensor networks. *Optim. Lett.* **7**(1), 157–172 (2013)
17. Moscibroda, T., Wattenhofer, R.: Minimizing interference in ad hoc and sensor networks. In: 2nd ACM SIGACT/SIGMOBILE International Workshop on Foundations of Mobile Computing (DIALM-POMC), pp. 24–33 (2005)
18. Raiconi, A., Gentili, M.: Exact and metaheuristic approaches to extend lifetime and maintain connectivity in wireless sensors networks. In: Pahl, J., Reinert, T., Voss, S. (eds.) *Network Optim. Lecture Notes in Computer Science*, vol. 6701, pp. 607–619. Springer, Heidelberg (2011)
19. Rossi, A., Singh, A., Sevaux, M.: An exact approach for maximizing the lifetime of sensor networks with adjustable sensing ranges. *Comput. Oper. Res.* **39**(12), 3166–3176 (2012)

20. Sterle, C., Sforza, A., Amideo, E.A., Piccolo, C.: A unified solving approach for two and three dimensional coverage problems in sensor networks. *Optim. Lett.* **10**(5), 1101–1123 (2016)
21. Wang, C., Thai, M.T., Li, Y., Wang, F., Wu, W.: Minimum coverage breach and maximum network lifetime in wireless sensor networks. In: *Proceedings of the IEEE Global Telecommunications Conference*, pp. 1118–1123 (2007)

# A Heuristic for Multi-attribute Vehicle Routing Problems in Express Freight Transportation

Luigi De Giovanni, Nicola Gastaldon, Ivano Lauriola  
and Filippo Sottovia

**Abstract** We consider a multi-attribute vehicle routing problem arising in a freight transportation company owning a fleet of heterogeneous trucks with different capacities, loading facilities and operational costs. The company receives short- and medium-haul transportation orders consisting of pick-up and delivery with soft or hard time windows falling in the same day or in two consecutive days. Vehicle routes are planned on a daily basis taking into account constraints and preferences on capacities, maximum duration, number of consecutive driving hours and compulsory drivers rest periods, route termination points, order aggregation. The objective is to maximize the difference between the revenue from satisfied orders and the operational costs. We propose a two-levels local search heuristic: at the first level, a variable neighborhood stochastic tabu search determines the order-to-vehicle assignment, the second level deals with intra-route optimization. The algorithm provides the core of a decision support tool used at the planning and operational stages, and computational results validated on the field attest for an estimated 9% profit improvement with respect to the current policy based on human expertise.

**Keywords** Multi-attribute VRP · Tabu search · Express freight transportation

---

L. De Giovanni (✉) · N. Gastaldon · I. Lauriola  
Dipartimento di Matematica “Tullio Levi-Civita”, Università di Padova,  
Via Trieste 63, 35121 Padova, Italy  
e-mail: luigi@math.unipd.it

N. Gastaldon  
e-mail: nicola.gastaldon@phd.unipd.it

I. Lauriola  
e-mail: ivanolauriola@gmail.com

F. Sottovia  
Trans-Cel, Via L. da Zara 33, 35020 Albignasego, Italy  
e-mail: info@trans-cel.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_17

# 1 Introduction

Vehicle routing Problems (VRPs) are a wide class of combinatorial optimization problems of both theoretical and practical interest, arising in transportation and logistics. In its basic version, the VRP considers a set of clients and a fleet of vehicles and asks for determining a set of routes, originating and terminating at a central depot, so that each client is visited by one vehicle and the total routes cost is minimized. With the aim of devising algorithms for real-world applications, one of the trends in recent research on VRPs is taking more and more attributes simultaneously into account: vehicle capacities, time windows, pickup and delivery, heterogeneous fleet, open routes, hours-of-service regulations etc. This gives rise to the so called Multi-Attribute (or Rich) VRP (MAVRP [3, 7]).

Among the works in the Operations Reserch literature devoted to MAVRPs, we cite the following. An exact approach suitable for many variants of the VRP is proposed in [1, 2], where a set partition formulation is solved by combining dual-ascent procedures and a cut-and-column generation algorithm. A real-world VRP involving several attributes (multiple capacities, hours-of-service regulations, open routes, split-delivery, client-vehicle compatibility constraints etc.) is solved in [4] through a column generation approach using a bounded bidirectional dynamic programming algorithm for the pricing problem. [6] and [7] overview several flexible heuristics able to adapt to the VRP definitions arising in different settings, and handle a variety of objectives and side constraints. Metaheuristic approaches, such as Tabu Search, Genetic Algorithms, Ant Colony Optimization etc., are very popular for solving MAVRPs [5]. The research presented in [8] hybridizes genetic algorithms and local search, and proposes a unified framework for solving a wide range of large-scale vehicle routing problems with time windows, route-duration constraints and further attributes related to client assignment.

We consider a MAVRP inspired by the daily operations at Trans-Cel, an express freight transportation company operating a fleet of heterogeneous vehicles to service short- and medium-haul transportation requests, consisting of pickups and deliveries to be fulfilled according to time windows falling within the same day or two consecutive days. Clients ask for a flexible just-in-time service and issue their requests fairly less than 24 h in advance. As a consequence, vehicle routes are planned on a daily basis and further re-optimizations are triggered by requests issued during routes operation. The scope of this paper is the development of an algorithm to support the operations manager during the daily and real-time assignment of requests to routes. The paper is organized as follows. Section 2 describes the problem, providing the details of its attributes. Section 3 reports the proposed solution approach, based on tabu search with multiple initial solutions, variable neighborhood, stochastic exploration and heuristic neighbor evaluation. Different variants of the solution algorithm have been tested on several real instances, and compared to the solution proposed by the operations manager: the results are shown in Sect. 4 together with some final remarks.

## 2 Problem Statement

The entities that define the MAVRP under study are positions, vehicles, orders and routes. Positions represent the addresses where pickups and deliveries take place. A special position is the depot, corresponding to the main facility of the transportation company. For each ordered pair  $(i, j)$  of positions, a distance  $d_{ij}$  is given. We consider a heterogeneous fleet of vehicles: each vehicle has its own capacity (volume, weight), loading facilities (e.g., tail lift, side door, crane) and operational costs (fixed deployment cost and a cost per unit of distance). We remark that capacity relates to on board freight (deliveries make capacity available again).

Orders represent the client's pickup and delivery requests. Each order specifies pickup and delivery positions, time windows within which pickup and delivery have to start, service times, size (volume and weight) and value. Notice that an order may have multiple deliveries, meaning that more delivery operations are specified for the same pickup, and all of these operations have to be performed by the same vehicle. The time windows can be hard (the operation must be performed within the time limits) or soft (we pay a penalty proportional to the time window violation): however, in case a vehicle arrives early, it waits until the beginning of the time window. A priority is associated to each order with three possible values: mandatory (must be fulfilled), and urgent and normal, with different penalties in case they are not executed. Finally, each order may specify the required loading facilities, if any, affecting the set of vehicles it can be assigned to. For some *fixed orders*, the assignment to an individual vehicle is a-priori fixed. We remark that the pickup and delivery dates of an order fall in the same day (*same-day order*) or in two consecutive days (*next-day order*). As a consequence the *daily orders* relevant for a specific day  $d$  are the same-day and the next-day orders of  $d$  (the last leaving pending deliveries for day  $d + 1$ ), and the next-day orders of day  $d - 1$  (pending deliveries for  $d$ ). Related operations are called *today* (resp. *tomorrow*) *operations* if their date is  $d$  (resp.  $d + 1$ ).

A route is the daily sequence of pickups and/or deliveries to be executed by an individual vehicle. Routes are normally open: a route starts at the position in which the vehicle has been positioned the day before, and it ends at the position of the last operation. In some cases a route may be forced to end at the depot (e.g. for maintenance issues). The design of a route is subject to hours-of-service regulations that introduce a compulsory break of at least 45 min after at most 4 h and 30 min of total driving time, and a compulsory night break of at least 9 h after either 9 h of total driving time or 13 h of total working (driving or waiting for pickup and delivery operations) time.

Further attributes are related to preferences on assigning subsets of orders to the same route (e.g. orders involving a same client), maximum route duration (e.g. due to drivers' requests or used to balance the long-term drivers' workload), route ending position, order assignment (e.g. preferred truck facilities). Also, constraints on order precedences may exist, as well as on the minimum number of vehicles that must be available at the depot at the end of the day.

Summarizing, given the set of daily orders and the initial vehicle positions, we want to determine an optimal set of orders to be executed and a set of feasible routes through them. The objective is to maximize profit and preference matching, the profit being defined as the difference between the total value of the executed orders, and the costs associated to vehicle operations and penalties for missed orders and time windows violation. Missed preferences are penalized in the objective function.

### 3 Two-Levels Tabu Search Approach

The operations manager has to solve the problem described above on a daily basis, before the deployment of vehicles on their routes. Moreover, the company receives further orders during the operations of the initially planned routes, and a re-optimization is needed starting from the current vehicles status (positions and on board orders), which is constantly available through the fleet management system. The algorithm described in this section aims at supporting the operations manager during the initial and the real-time route optimization. Due to the inner complexity of the problem and the operational scenarios it supports, a meta-heuristic approach has been devised.

As usual for MAVRPs, three main orders of decision have to be made: the assignment of orders to vehicle routes, the sequencing of orders, and the computation of the best path through the order positions. For the sake of algorithm design, we decompose the problem into two levels: the first level determines the order-to-route assignment, the second level deals with operations sequencing and best path computation, that is, intra-route optimization.

For the first level, we propose a heuristic based on Tabu Search: it starts from an initial solution (current solution) and it generates a set of neighbor solutions by applying some perturbation to the current one; the solutions are evaluated according to a score function and the best neighbor is chosen as the new current solution; the process iterates until a termination condition occurs (e.g. maximum number of iterations). The incumbent solution is updated each time a better current solution is generated. In order to avoid cycling, a tabu list is maintained, storing information on the last visited solutions, so that they are excluded from eligible neighbors. In order to adapt this very general schema to our problem, we need to define the following components: the score function to evaluate solutions, initial solution, neighborhood and tabu list.

#### Solution Evaluation and Intra-route Optimization

According to the chosen problem decomposition, the Tabu Search explores the space of all the possible assignments of orders to vehicle routes. Each solution is evaluated by summing up the values of assigned orders and subtracting the following generalized cost function ( $r$  indexes the routes included in the solution):

$$\sum_r [A(r) + w_B B(r) + C(r) + w_D D(r) + w_E E(r)] + F + w_G G + w_H H + \sum_{\rho \in \{I, J, K, L\}} w_\rho \rho$$



- $A(r)$  is the actual vehicle cost: it includes the fixed deployment cost and the variable cost depending on the length of  $r$  up to the last today operation;
- $B(r)$  is the prospective vehicle cost: it is the variable cost depending on the length of  $r$  between the last today and tomorrow operations. This cost component and the related weight aim at better selecting the final route positions and drive the search towards solutions accounting for the next-day costs;
- $C(r)$  is the cost due to penalties for soft time windows violations on  $r$ ;
- $D(r)$  is the amount of hard time windows violations on  $r$ ;
- $E(r)$  is the time of  $r$  exceeding the working time threshold;
- $F$  sums up the penalties associated to missed urgent and normal priority orders;
- $G$  is the amount of missed mandatory orders;
- $H$  counts the missing vehicles at the depot with respect to the minimum required;
- $I, J, K, L$  account for the penalties associated to violated preferences on, respectively, same route orders, route duration, ending positions and order assignment;
- $w_X$  is the weight (to be calibrated) of the performance indicator  $X$ . Non-weighted components are estimated equivalent costs provided by the company.

Components  $D(r)$ ,  $E(r)$ ,  $G$  and  $H$  account for constraints violation: the proposed tabu search allows visiting unfeasible solutions, which are suitably penalized.

We observe that, in order to evaluate most of the indicators above, the sequence of operations must be determined, which corresponds to the second-level decisions, that is, the intra-route optimization. We perform this task by a second-level heuristic based on local search. It receives the orders to be inserted in the route from the top level calling procedure (namely the first-level initial constructive heuristics and neighborhood functions that will be described in the following). Each time a new order  $o$  is inserted, the intra-route optimization runs the following steps:

1. set an initial position for  $o$  by evaluating (according to the score function defined above) the best possible insertion of all the *today* operations of  $o$  between two consecutive *today* operations already in the route;
2. if  $o$  is fixed, perform a local search by generating neighbor solutions from inserting any *today* operation between each pair of *today* operations and moving to the best neighbor if it improves over the current solution;
3. perform a local search by generating neighbor solutions obtained from swapping the position of any pair of *today* operations and moving to the best neighbor if it improves over the current solution;
4. complete the sequence by inserting *tomorrow* operations according to Nearest Neighbor heuristic.

Notice that the local search moves and the insertions are discarded if they violate any constraint other than the ones penalized in the score function.

### Initial Solution

Two constructive heuristics are proposed to build the initial solution. The *Round-robin insertion with Priority* (RP) heuristic defines three classes of vehicles: vehicles with orders on board (they must be deployed), vehicles with starting position

different from the depot, and initially empty vehicles at the depot. Vehicles within the same class are sorted by ascending operational costs. Orders are assigned to vehicles in the first class, one order at a time for each vehicle in a round-robin fashion. To this end, orders are sorted by a proximity criterion: given a route  $r$ , the score associated to an order  $o$  positioned in  $j$  is  $p_r(j) = R(o) + d_{0j} - \min_{i \in Q(r)} \{d_{ij}\}$ , where  $Q(r)$  is the set of positions in  $r$ , 0 is the depot, and  $R(o)$  is a parameter depending on the order priority. The ratio is to prefer the early (and hence hopefully better) insertion of priority orders and orders far away from the depot. Once the capacities of the vehicles in the first class are saturated, remaining orders are assigned to the second class according to the same procedure. The assignment of remaining orders occurs sequentially (a vehicle is saturated before considering the following one) for the third class. In such a way, we try to accommodate as much priority orders as possible in different routes, while saving the deployment of some vehicles.

The *Best Insertion* (BI) heuristic sorts the orders by descending distance from the depot, and each order is inserted in the vehicle route optimizing the score function of the partial solution under construction.

Notice that an insertion triggers the execution of the first-level heuristic for intra-route optimization and it is discarded if it yields the violation of any constraints other than the ones penalized in the score function (in case of fixed orders, a warning is issued). All the unassigned orders are collected in a dummy route assigned to a dummy vehicle.

### Tabu Search Neighborhoods and Exploration Strategy

The following inter-route moves are defined to perturb the solutions in the first-level tabu search and explore the space of order-to-vehicle assignments (we remark that only non-fixed orders are involved):

- *Single order relocation* (1R): an order is moved from its current route to a new route. The size is  $O(n \cdot v)$ ,  $n$  and  $v$  being the number of non-fixed orders and routes, respectively;
- *Two-orders swap* (2S): given two orders in two different routes, their assignment is swapped. The size is  $O(n^2)$ ;
- *Two-orders relocation* (2R): two orders assigned to a route  $r_1$  are moved to a route  $r_2$ . The size is  $O(n^2 \cdot v)$ ;
- *Two-orders chain shift* (2C): an order  $o_1$  assigned to route  $r_1$  is moved to a route  $r_2$ , and an order  $o_2$  assigned to  $r_2$  is moved to a third route  $r_3$ . The size is  $O(n^2 \cdot v)$ .

Evaluating a move involves the intra-route optimization procedure, to be called once for each affected route. The complexity of the proposed neighborhoods depends on the number of affected routes: two for 1R, 2S and 2R, three for 2C. For the sake of efficiency, a granular exploration of 2C is proposed, discarding moves if they involve nodes at a distance greater than a threshold  $Z$ .

The neighborhoods are sorted as listed above and explored in a variable neighborhood fashion: at each iteration a neighborhood is explored if and only if the previous one does not provide an improving solution, and the best solution from any of the

explored neighborhoods is taken as the new current solution. A stochastic exploration of the search space is also devised: in this case the new current solution is chosen at random among the best five generated by any explored neighborhood, with probability proportional to the score function. The tabu list stores the last  $T$  moves yielding selected current solutions, as to forbid inverse moves. The first-level tabu search stops as soon as a maximum total number  $M_1$  of iterations or a maximum number  $M_2$  of iterations without improvement is reached.

### 4 Computational Results and Conclusions

The algorithm has been implemented in C++ (Microsoft compiler) and run on an Intel Core i5-5200 2.20 GHz CPU, with 8 GB RAM. During a first phase, the algorithm has been trained on several instances by analyzing the results in the light of the operations manager’s solutions, and by discussing its features with the company planning team, thus yielding the calibration of its parameters (the weights in the score function and the tabu search parameters). In particular, we set  $Z = 100$  km,  $M_1 = 500$ ,  $M_2 = 50$  and  $T$  between 4 and 8, depending on the number of operations of non-fixed orders. During a second phase, the calibrated algorithm has been compared to the operations manager’s solution on a set of 30 instances related to 30 working days in March and April 2016. Instances include from 8 to 46 orders (25.6 on average) and from 17 to 95 pickup and delivery operations (60.1 on average). The number of operations for non-fixed orders ranges from 5 to 61 (22.0 on average), and the ones in fixed orders range from 12 to 54 (38.1 on average). The results are summarized in Table 1. Rows are devoted to different algorithm configurations: RP and BI refer to running only one initial heuristic; BestH runs the two heuristics and chooses the best solution; RP+DET (resp. BI+DET) runs the deterministic version of the Tabu Search starting from the initial RP (resp. BI) solution; RP+RND (resp. BI+RND) runs three repetitions of the stochastic version of the Tabu Search

**Table 1** Computational results for different algorithm configurations

Algorithm	Improvement (%)			Running time (s)			Win (%)	Dom. (%)
	Avg	Min	Max	Avg	Min	Max		
RP	-10.5	-42.3	5.9	1.2	0.0	10.9	-	-
BI	7.3	-10.2	15.3	0.1	0.0	0.3	-	-
BestH	7.3	-10.2	15.3	1.3	0.0	11.1	-	-
RP+DET	8.5	-3.9	15.3	9.6	0.0	53.5	53.3	16.7
RP+RND	8.4	-2.7	15.3	19.8	0.0	146.8	50.0	10.0
BI+DET	9.3	2.2	15.3	33.2	0.0	295.0	76.7	23.3
BI+RND	9.1	-1.3	15.3	72.6	0.0	528.7	63.3	6.7
BestTS	9.6	2.2	15.3	172.3	0.1	551.0	100.0	100.0

starting from the initial RP (resp. BI) solution; BestTS runs all the previous Tabu Search configurations and return the best solution. Columns report: the configuration name; the percent improvement  $(A - B)/A$  where  $A$  is the algorithm solution and  $B$  is the operations manager's solution (average, minimum and maximum), the running time in seconds (average, minimum and maximum), the frequency the configuration finds a solution within 1% of the best solution (provided by BestTS), the frequency the configuration dominates all the others. The overall algorithm, including multiple starts and neighborhood exploration strategies, is able to provide an average 9.6% improvement over the operations manager's solution within about 3 min (10 in the worst case). On average, all the tested Tabu Search configurations improve over the baseline by 8.4 to 9.3%, even if, for some configurations, the operations manager still finds better solutions for some instances. The Tabu Search starting from BI heuristic and deterministically exploring the variable neighborhood seems to better trade-off efficiency, effectiveness and reliability, always providing solutions better than the baseline, with an improvement between 2.2 and 15.3% (9.3% on average) obtained within 33.2 s on average (5 min in the worst case). The last column shows that all configurations reach a strictly better solution in some instances. Notice that all the algorithms but RP give the same maximum improvement on a same medium-size instance where BI is able to find a good solution that cannot be further improved by tabu search. An analysis of the moves selected at each iteration shows that all the proposed neighborhoods play a significant role: as expected, the most selected moves are 1R (70.9% of times); 2S, 2C and 2R follow in the given order, selected the 15.4%, 8.3% and 5.4% of times, respectively.

The algorithm is currently integrated in the operations planning activities, and is used for both the initial daily planning and the real-time insertion of new orders: the initial solution is taken from the current fleet status, by fixing all the orders but the new ones, and the algorithm is able to suggest a solution within a few seconds. Also, the operations manager has the opportunity of performing what-if analyses: she/he changes the solution provided by the algorithm and reruns it with new fixed orders and precedence constraints. Further ongoing research aims at improving the efficiency of the algorithm through the implementation of incremental neighborhood evaluation, and the design of a more focused diversification phase.

## References

1. Baldacci, R., Bartolini, E., Mingozzi, A., Roberti, R.: An exact solution framework for a broad class of vehicle routing problems. *Comput. Manag. Sci.* **7**(3), 229–268 (2010)
2. Baldacci, R., Mingozzi, A.: A unified exact method for solving different classes of vehicle routing problems. *Math. Program.* **120**(2), 347–380 (2009)
3. Caceres-Cruz, J., Arias, P., Guimarans, D., Juan, A.A.: Rich vehicle routing problem: survey. *ACM Comput. Surv.* **47**(2), Article 32, 28 pp. (2014)
4. Ceselli, A., Righini, D., Salani, M.: A column generation algorithm for a rich vehicle routing problem. *Transp. Sci.* **43**(1), 56–69 (2009)
5. Gendreau, M., Potvin, J.Y., Bräysy, O., Hasle, G., Lokketangen, A.: Metaheuristics for the vehicle routing problem and its extensions: a categorized bibliography. In: Golden, B., Raghavan,

- S., Wasil, E. (eds.) *The Vehicle Routing Problem: Latest Advances and New Challenges*, pp. 143–169. Springer, New York (2008)
6. Laporte, G., Ropke, S., Vidal, T.: Heuristics for the vehicle routing problem. In: Toth, P., Vigo, D. (eds.) *Vehicle Routing: Problems, Methods, and Applications*, pp. 87–116. MOS-SIAM Series on Optimization (2014)
  7. Vidal, T., Crainic, T.G., Gendreau, M., Prins, C.: Heuristics for multi-attribute vehicle routing problems: a survey and synthesis. *Eur. J. Oper. Res.* **231**(1), 1–21 (2013)
  8. Vidal, T., Crainic, T.G., Gendreau, M., Prins, C.: A unified solution framework for multi-attribute vehicle routing problem. *Eur. J. Oper. Res.* **234**(3), 658–673 (2013)

# Global Optimization Procedure to Estimate a Starting Velocity Model for Local Full Waveform Inversion

Bruno Galuzzi, Elena Zampieri and Eusebio Stucchi

**Abstract** Finding an efficient procedure to solve a seismic inversion problem, such as Full Waveform Inversion, still remains an open question. This is mainly due to the non linearity of the misfit function, characterized by the presence of multiple local minima, that cause unsuccessful results of local optimization strategies when the starting model is not in the basin of attraction of the global minimum. Therefore, the use of a global optimization strategy in order to estimate a suitable starting velocity model for Full Waveform Inversion is a crucial point. In this work we propose a new method which is based on the application of the Adaptive Simulated Annealing algorithm using a coarse inversion grid, different from the domain modelling one, that allows the reduction of the number of unknowns. Numerical results show that the application of our strategy to an acoustic Full Waveform Inversion provides a good starting model for a local optimization procedure, lying in the basin of attraction of the global minimum.

**Keywords** Simulated annealing · Seismic inversion · Acoustic wave equation

---

B. Galuzzi (✉) · E. Stucchi  
Department of Earth Sciences, University of Milan,  
Via Cicognara 7, 20129 Milan, Italy  
e-mail: bruno.galuzzi@unimi.it

E. Stucchi  
e-mail: eusebio.stucchi@unimi.it

E. Zampieri  
Department of Mathematics, University of Milan,  
Via Saldini 50, 20133 Milan, Italy  
e-mail: elena.zampieri@unimi.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_18

# 1 Seismic Exploration and Inversion

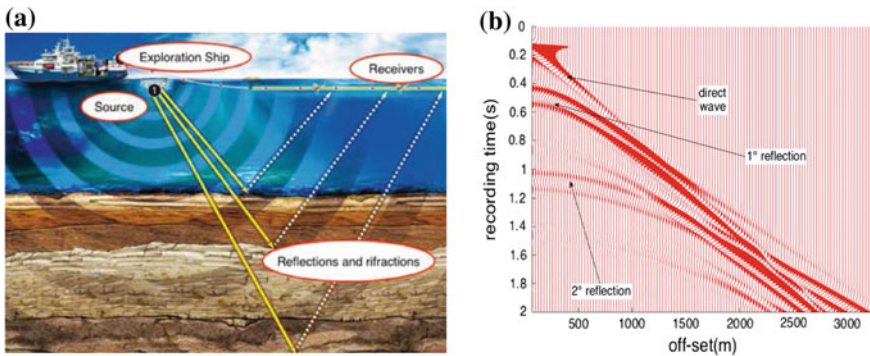
## 1.1 Seismic Tomography

Seismic exploration estimates the geological properties of the subsurface using techniques based on recording, processing and studying artificially induced seismic waves [10]. The source consists of a device that generates controlled seismic energy. Once it is activated, the receivers, located on the surface at increasing distance from the source, record the waveforms of the seismic events caused by the subsurface rock discontinuities. Figure 1a shows a sketch of a 2D marine seismic acquisition. The acquired data of a single shot is called seismogram and is formed by the recording of all the receivers, ordered as a function of the distance from the source, that is called offset. Figure 1b shows an example of seismogram, highlighting the traveltimes of the main seismic events.

The aim of seismic tomography is to estimate the elastic properties, that explain the events observed in a seismogram (reflections, refractions, diffractions). It can be formulated as an inversion problem:

$$m = H(d_{obs}), \quad (1)$$

where the observed data  $d_{obs}$  are one or more seismograms obtained during a seismic acquisition,  $m = m(\vec{x})$  is the model of the elastic properties of the Earth, and  $H$  is a non-linear inversion operator. In this work we consider the Full Waveform Inversion (FWI), that is a recently developed technique to solve this problem, exploiting the full waveform of the recorded seismic events.



**Fig. 1** **a** Sketch of a marine seismic acquisition; **b** A seismogram given by the recording of all the receivers of a single shot

## 1.2 Full Waveform Inversion (FWI)

The basic idea of FWI is to formulate the inversion problem as a minimization one [13, 14]: find  $\bar{m} \in M$  such that

$$F(\bar{m}) = \min_{m \in M} F(m), \quad (2)$$

where  $M$  is the set of all possible geological models and  $F(m) \geq 0$  is a misfit function

$$F(m) = ||d_{pred}(m) - d_{obs}||, \quad (3)$$

that measures the difference between the observed  $d_{obs}$  and the predicted seismogram  $d_{pred}(m)$ , obtained by the numerical solution of the wave equation operator  $G$  applied to a model  $m$ :

$$d_{pred} = G(m), \quad (4)$$

with  $G \approx H^{-1}$ . A classical function used in the contest of FWI is the  $L^2$ -norm difference between the observed and the synthetic seismograms

$$F(m) = \frac{1}{2} \sum_{r=1}^{n_r} \int_0^T [d_{pred}(t, \vec{x}_r, m) - d_{obs}(t, \vec{x}_r)]^2 dt, \quad (5)$$

where  $\vec{x}_r \in X^r$  and  $n_r$  represent the positions and the number of receivers, respectively, and  $T$  is the recording time. The solution of this problem depends both on the type of wave equation operator and on the numerical scheme adopted to approximate the propagation of the seismic waves.

## 1.3 Acoustic FWI in Case of FD Scheme

If we consider in the geological model only the spatial distribution of the  $P$ -wave velocity  $v(\vec{x})$ , the generation and propagation of seismic wave is modeled by the 3D acoustic wave equation [1]:

$$\ddot{p}(\vec{x}, t) - v(\vec{x})^2 \Delta(\vec{x}, t) = \delta(\vec{x} - \vec{x}_0) s(t), \quad (6)$$

where  $t \in [0, T]$  is the recording time,  $\vec{x} \in D \subset \bar{R}^3$  is the space domain,  $p$  is the acoustic pressure of the wave,  $\vec{x}_0$  is the location of the source, and  $s$  is the seismic wavelet. The predicted seismograms  $d_{pred}$  correspond to the restriction on  $X^r$  of the solution of the acoustic wave equation:

$$d_{pred}(t, \vec{x}_r, v) = p(\vec{x}, t)|_{\vec{x}_r \in X^r}. \quad (7)$$



Then the FWI problem in (2) becomes an acoustic optimization problem:

$$\min_{v \in V} F(v) = \min_{v \in V} \left( \frac{1}{2} \sum_{r=1}^{n_r} \int_0^T [d_{pred}(t, \vec{x}_r, v) - d_{obs}(t, \vec{x}_r)]^2 dt \right) \quad (8)$$

with  $V$  the set of all possible  $P$ -wave velocity models.

The approximation of Eq. (6) using finite differences [11] can be performed by sampling  $D$  with a uniform step  $dx$ , obtaining a regular grid  $D^{i,j,l}$ , formed by  $n_x \cdot n_y \cdot n_z$  grid nodes, with  $i = 1, \dots, n_x$ ,  $j = 1, \dots, n_y$  and  $l = 1, \dots, n_z$ . The velocity model must be sampled on the grid nodes  $v^{i,j,l} = v(x_i, y_j, z_l)$  as well.

According to this parametrization, the acoustic FWI problem in (8) can be approximated by an optimization problem with the number of unknowns given by the number of nodes in the modeling grid

$$\min_{v \in V} F(v) \approx \min_{v^{i,j,l} \in V^{i,j,l}} F(v^{i,j,l}) \quad (9)$$

where  $V^{i,j,l}$  represents the set of the  $P$ -wave velocity models discretized on the grid  $D^{i,j,l}$ .

## 2 Global FWI Using ASA and Two Grid Methods

### 2.1 Iterative Procedures for FWI

Since  $F(m)$  is generally a complicated non-linear function of the model  $m$ , iterative minimization procedures [7] are normally used to attain the global minimum  $\bar{m}$  of  $F$ . Starting from a plausible initial model  $m_0$ , iterative minimization updates the current model  $m_k$  to a new model  $m_{k+1}$ , given by:

$$m_{k+1} = m_k + \gamma_k h_k \quad (10)$$

where  $F(m_{k+1}) < F(m_k)$  and  $h_k$  and  $\gamma_k \in \bar{R}^+$  are the descend direction and the step length, respectively. A local optimization procedure, such as the steepest descend or the conjugate gradient method, under certain regular assumptions for the misfit function, converges to the local minimum  $\bar{m}_{loc}$  in the basin of attraction of the starting model  $m_0$  [7]. Unfortunately, seismic inverse problems, and in particular FWI, are characterized by the presence of multiple local minima due to cycle-skipping effect caused by the oscillatory behaviour of the seismic data.

To avoid the convergence towards local minima, it is necessary to estimate a starting model  $m_0$  that is in the basin of attraction of the global minimum  $\bar{m}$ . Usually, when the number of unknowns is very high, as in the FWI case, parallel algorithms are preferred [8]. However, in this work we propose to use the Adaptive Simulated

Annealing (ASA), which is an algorithm that shows a good rate of convergence in many test global optimization problems when the number of unknowns is limited [9]. To reduce the high number of unknowns in the FWI, a coarse grid, different from the modelling grid, is used for the inversion problem [5, 6].

### 2.2 Adaptive Simulated Annealing

The simulated annealing (SA) [4] is a stochastic optimization method that models the metallurgical annealing process by using the concepts of cooling and heating, creating an iterative sequence of models  $\{m_k\}_{k \geq 0}$ . Figure 2 shows a sketch of an iterative minimization, using the SA. For each iteration  $k$ , a new candidate model is generated in a neighborhood of the current model using the generation formula

$$y = m_k + \Delta m_k(T_{g,k}, p), \tag{11}$$

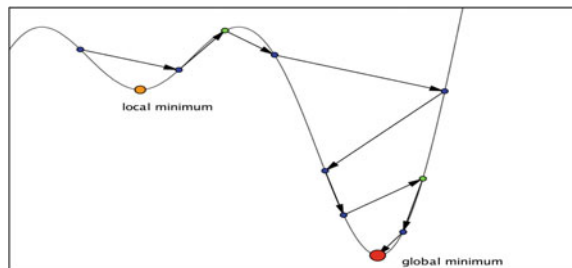
where  $\Delta m_k$  is the step size, depending on a parameter  $T_{g,k} = T_g(k) > 0$ , called generation temperature, and  $p$  is a random uniform distributed numbers in  $[0, 1]$ . The step size is proportional to  $T_g$ . Once the candidate model  $y$  is created, the algorithm chooses whether the model is accepted or not, according to the following formula:

$$m_{k+1} = \begin{cases} y, & \text{if } p < \min(1, e^{-\frac{f(y)-f(x_k)}{T_{a,k}}}) \\ m_k, & \text{otherwise} \end{cases} \tag{12}$$

where  $T_{a,k} = T_a(k) > 0$  is a parameter called acceptance temperature. The new model is always accepted if the new value of the objective function is lower, whereas if this value is higher, the model is accepted with a probability dependent on  $T_{a,k}$ .

At the early stages of the optimisation, the initial generation and acceptance temperatures ( $T_{a,0}, T_{g,0}$ ) are set to high values, for a wide exploration of the model space; subsequently, their values are progressively reduced to the final values ( $T_{a,f}, T_{g,f}$ ) to

**Fig. 2** An example of an iterative minimization sequence, generated by SA, to find the global minimum (red point). Note the climbing point (green point)



focus the search on the most promising zones of the model space. In this work we used a variant of SA, called adaptive simulated annealing (ASA) [3], that uses different generation formulas for each  $i$ -th direction of the model space:

$$m_{k+1}^i = m_k^i + \Delta m_k^i(T_{g,k}^i, p^i), \quad i = 1, \dots, n \quad (13)$$

where  $n$  is the dimension of the model space,  $T_{g,k}^i$  are the generation temperatures,  $\Delta m_k^i$  are the step sizes and  $p^i$  are random numbers uniformly distributed over  $[0, 1]$ . The different step sizes are obtained using the formula:

$$\Delta y_k^i = \text{sgn}\left(p^i - \frac{1}{2}\right) \left( \left[ 1 + \frac{1}{T_{g,k}^i} \lfloor 2^{p^i-1} \rfloor \right] - 1 \right) T_{g,k}^i. \quad (14)$$

The generation and acceptance temperatures are characterized by an exponential decrease with respect to the number of iterations  $k$  and of accepted models  $k_a$ , respectively:

$$\begin{cases} T_{g,k+1}^i = T_{g,0}^i e^{-c_i \sqrt[k]{k}}, \\ T_{a,k+1} = T_{a,0} e^{-c_a \sqrt[k]{k_a}} \end{cases} \quad (15)$$

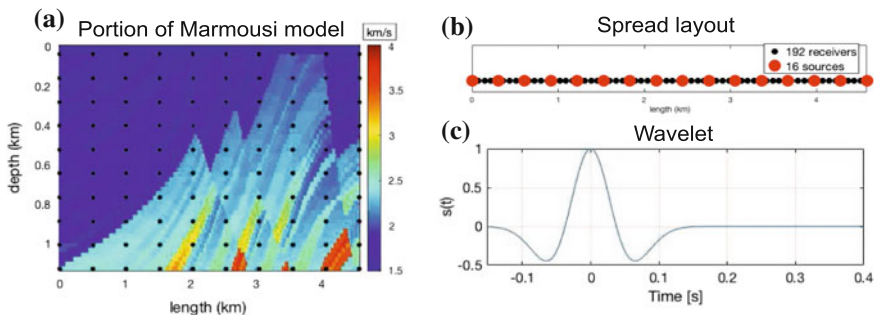
where  $c_i$  and  $c_a$  are scalability factors. Both  $T_{g,k}^i$  and  $T_{a,k}$ , after a given number of iterations  $n_{iter}$ , reach their final values and remain constant.

According to the concept of local adaptability and to reduce the possibility of becoming trapped in a local minimum, both the generation and the acceptance temperature can be increased again (re-annealing) as a function of  $k$  and  $k_a$ , respectively.

### 2.3 Two Grid Methods

Stochastic optimization methods require expensive computational time when applied to inverse problems characterized by large dimension of the model space [9]. Unfortunately, the number of possible unknowns for a FWI can be very high. For instance, in case of acoustic FWI using a finite difference approximation of Eq. (6), the unknowns are the values of the  $P$ -wave velocity on the nodes of the modelling grid and can be as many as some thousand.

To address this issue, we can describe the geophysical properties of the subsurface model on a coarse grid, thus reducing the number of unknowns. The coarse grid can be a subset of the modelling grid where the number and the distribution of the nodes is settled as a function of the accuracy's degree. The transition from the coarse to the modelling grid can be fulfilled through an interpolation procedure. We use the ASA algorithm to estimate a smooth starting velocity model for a local inversion in an affordable computational time.



**Fig. 3** **a** The true velocity and the coarse grid, represented by the black dots. **b** The recording spread formed by 16 sources and 192 receivers. **c** The source wavelet

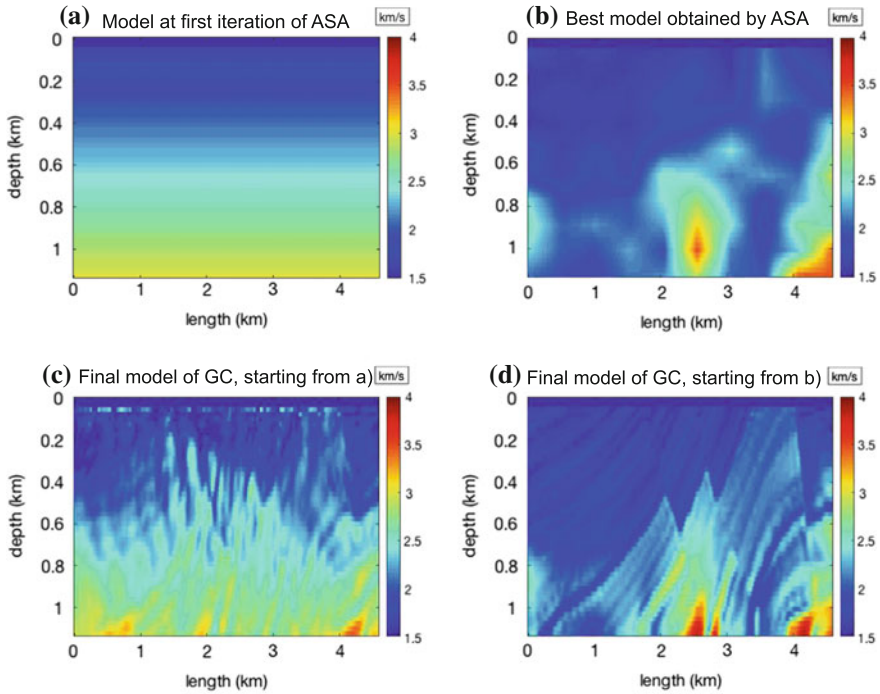
### 3 Example of Application on the Marmousi Model

We test the inversion procedure on a portion of the Marmousi model (see Fig. 3a) [12]. In order to obtain the observed data, we simulate a marine seismic acquisition consisting of 16 different seismic shots recorded by a spread of 192 receivers, equally spaced 24 m apart (see Fig. 3b). Both sources and receivers are at a depth of 24 m. To simulate the seismograms we implemented the 2D acoustic wave equation (6) and computed the solution on the receiver nodes. Source wavelet is a Ricker wavelet with peak frequency of 6 Hz and maximum frequency of 18 Hz (see Fig. 3c). In the numerical implementation we set  $dt = 0.002$  s (sampling time),  $T = 3$  s (recording time), and  $dx = 24$  m (sampling space). The modelling grid is made of 9216 grid nodes, with  $n_x = 192$  and  $n_z = 48$ ; the water layer is modelled by the first two rows of the grid and remains fixed during the inversion. The coarse grid, indicated by the black dots in Fig. 2a, is composed of 100 nodes and we use a bilinear interpolation to bring the velocity model from the coarse to the modelling grid.

To apply the ASA algorithm, we set  $n_{iter} = 1000$ ,  $T_{g0}^i = 100$  and  $T_{gf}^i = 10^{-18} \forall i = 1, \dots, 100$ . Besides, we consider  $T_{a0} = \frac{1}{5} \sum_{i=1}^5 F(m_i)$ , with  $m_i$  random models, and  $T_{af} = 10^{-18}$ .

Figure 4a shows the 1D starting model we chose for the ASA algorithm, while Fig. 3b shows the best model obtained after 100000 iterations. We used these two models as starting point for two different local optimization procedures on the fine grid along that make use of the conjugate gradient method (CG) [2]. In Fig. 4c and d we show the final models obtained after 3000 iterations respectively.

The outcome of the first local optimization procedure in Fig. 4c is quite different from the true model in Fig. 3a. This means that a local optimization alone can give a solution that does not correspond to the real geological model in the area on investigation and therefore some previous steps of velocity estimation must be accomplished before the local optimization is started. The second model is very similar to the true one, except in some areas near the lateral and the bottom boundaries, where



**Fig. 4** **a** The model at ASA's first iteration. **b** The best model obtained by ASA after 30000 iterations. **c** The final model obtained through CG algorithm, starting from **(a)**. **d** The final model obtained through CG algorithm, starting from **(b)**

the seismic illumination is poor. This means that the starting model of the local procedure, estimated by the ASA algorithm on the coarse grid, is very close to—or in the basin of attraction of—the global minimum.

## 4 Conclusions

Using a global optimization procedure, consisting of a stochastic optimization algorithm and a two-grid approach, it is possible to estimate a good starting velocity model for a local optimization algorithm, that allows to solve seismic inversion problems. We test the procedure in the case of an acoustic Full Waveform Inversion, carried out on a portion of the Marmousi model. The good correspondence between the true and the final model obtained makes this procedure of particular interest especially in seismic inversion problems which are characterized by highly non-linearity and multiple local minima.

## References

1. Fichner, A.: Full Seismic Waveform Modelling and Inversion. Springer Science & Business Media (2010)
2. Fletcher, R., Reeves, C.M.: Function minimization by conjugate gradients. *Comput. J.* (1964). <https://doi.org/10.1093/comjnl/7.2.149>
3. Ingber, L.: Very fast simulated re-annealing. *Math. Comput. Modell.* (1989). [https://doi.org/10.1016/0895-7177\(89\)90202-1](https://doi.org/10.1016/0895-7177(89)90202-1)
4. Kirkpatrick, J.S., Gelatt, C.D., Vecchi, M.P.: Optimisation by simulated annealing. *Science* (1983). <https://doi.org/10.1126/science.220.4598.671>
5. Galuzzi, B., Tognarelli, A., Stucchi, E., Mazzotti, A.: Stochastic FWI on Wide-angle land data with different order of approximation of the 2D acoustic wave equation. In: 78th EAGE Conference and Exhibition (2016). <https://doi.org/10.3997/2214-4609.201601189>
6. Mazzotti, A., Bienati, N., Stucchi, E., Tognarelli, A., Aleardi, M., Sajeve, A.: Two-grid genetic algorithm full-waveform inversion. *Leading Edge* (2016). <https://doi.org/10.1190/tle35121068.1>
7. Nocedal, J., Wright, S.: Numerical Optimization. Springer (2006)
8. Sajeve, A., Aleardi, M., Stucchi, E., Bienati, N., Mazzotti, A.: Estimation of acoustic macro models using a genetic full-waveform inversion: applications to the Marmousi model. *Geophysics* (2016). <https://doi.org/10.1190/geo2015-0198.1>
9. Sajeve, A., Aleardi, M., Galuzzi, B., Stucchi, E., Spadavecchi, E., Mazzotti, A.: Comparing the performances of four stochastic optimisation methods using analytic objective functions, 1D elastic full-waveform inversion, and residual static computation. *Geophys. Prospect.* (2017). <https://doi.org/10.1111/1365-2478.12532>
10. Sherif, R.E., Geldart, L.P.: Exploration Seismology. Cambridge University Press. (1995)
11. Strickwerda, J.C.: Finite Difference Schemes and Partial Differential Equations. The Wadsworth & Brooks/Cole Mathematics Series (1989)
12. Versteeg, R.: The Marmousi experience: velocity model determination on a synthetic complex data set. *Leading Edge* (1994). <https://doi.org/10.1190/1.1437051>
13. Virieux, J., Operto, S.: A strategy for nonlinear elastic inversion of seismic reflection data. *Geophysics* (1986). <https://doi.org/10.1190/1.1442046>
14. Tarantola, A.: An overview of full waveform inversion in exploration geophysics. *Geophysics* (2009). <https://doi.org/10.1190/1.3238367>

# Ant Colony Optimization Algorithm for Pickup and Delivery Problem with Time Windows

M. Noubissi Tchoupo, A. Yalaoui, L. Amodeo, F. Yalaoui and F. Lutz

**Abstract** This paper presents an efficient meta-heuristic for the Pickup and Delivery Problem with Time Windows (PDPTW) based on Ant Colony Optimization coupled to dedicated local search algorithms. The objective function is the minimization of the number of vehicles and the minimization of the total distance travelled. In PDPTW, the demands are coupled and every couple is a request which must be satisfy in the same route. Thus, the feasible solution space is tightly constraint and then makes the design of effective heuristics more difficult. Experimental results on 56 instances of 100 customers of Li and Lim's benchmark show that the ACO coupled with PDPTW dedicated local search algorithms outperform existing algorithms. It returns in 98.2% (55/56) of cases a solution better or equal to the best known solution, and find a better solution than the best know in 44.6% (25/56).

## 1 Introduction

The Pickup and Delivery Problem with Time Windows (PDPTW) can be described as the design of a least cost routing plan to satisfy a set of transportation requests by a given identical vehicle fleet. Each request consists of delivering goods from a predefined location (pickup customer) to another one (delivery customer). In this

---

M.N. Tchoupo (✉) · A. Yalaoui (✉) · L. Amodeo (✉) · F. Yalaoui (✉)  
University of Technology of Troyes, LOSI ICD UMR, CNRS, 6281 Troyes, France  
e-mail: moise.noubissitchoupo@utt.fr

A. Yalaoui  
e-mail: alice.yalaoui@utt.fr

L. Amodeo  
e-mail: lionel.amodeo@utt.fr

F. Yalaoui  
e-mail: farouk.yalaoui@utt.fr

F. Lutz  
Hospital of Troyes, 101 Av. Anatole France, Troyes, France  
e-mail: frederic.lutz@hcs-sante.fr

problem, the routing plan is designed such that all vehicles start and end at the depot. The amount of goods must not exceed the vehicle's capacity. Each customer must be serviced within a given time windows. The service time indicates how long it will take for the pickup or delivery to be performed. For each request, the corresponding pickup customer must be visited before the corresponding delivery customer by the same vehicle and in the same route but not necessary immediately after. A vehicle is allowed to arrive at a location before the beginning of its time windows, and in this case must wait until the start of the time window. The problem is NP-hard as it contains the Travelling Salesman Problem with Time Windows (TSPTW) (See Dumas et al. [3]).

Numerous study have been done in PDPTW with the objective to minimize the number of vehicle (primary objective) and the total travelled distance (secondary objective). A simulated annealing with tabu search was proposed by Li and Lim [5] to solve PDPTW. Bent and Van Hentenryck [2] proposed a two-stage hybrid algorithm for PDPTW, where in the first stage the number of vehicles is decrease, while in the second stage the total travel cost is minimized by a Large Neighbourhood Search algorithm (LNS). An adaptive large neighbourhood search heuristic was proposed by Ropke and Pisinger [9]. Nagata and Kobayashi [6] successfully applied a Guided Ejection Search Algorithm to PDPTW. Nalepa et al. [7] proposed a parallel guided ejection search algorithm to solve PDPTW. In their approach, parallel processes co-operate periodically to enhance the quality of results and to accelerate the convergence of computations.

Tchoupo et al. [11] developed a Bender's decomposition algorithm for PDPTW with heterogeneous fleet (HVRPPDTW) to minimize the hierarchical objective. In the homogeneous case, their proposed approach was able to solve optimally instances up to 100 demands in reasonable computational time. To our knowledge, this method is the only exact algorithm for the PDPTW with hierarchical objective. For a survey on pickup and delivery problems see [8].

In the state of the art, there are not effective constructive heuristic to solve PDPTW. Indeed, the studies used iterative methods based on insertion and remove of requests. The greatest challenge in a constructive method is to find fast heuristics to choose the next demand to satisfy and verify there exists a path to achieve all delivery demands in current vehicle, whose corresponding pickup demands are not yet satisfied. Finding a feasible path to serve a given set of delivery demands is equivalent to solve a Hamiltonian path problem (NP-complete). This issue had been previously identified by Dumas et al. in [3] when they proposed a labelling algorithm for the Elementary Shortest Path Problem with Time Windows, Capacity, and Pickup and Delivery (ESPPTWCPD). In a proposed labelling algorithm, they proposed to consider only the subsets of deliveries of cardinality one and two.

The model proposed by Goss et al. [4] to explain the foraging behaviour of ants was the main source of inspiration for the development of ant colony optimization. The ACO was applied successfully to solve Vehicle Routing Problem as done by Belmecheri et al. [1] to solve the Vehicle Routing Problem with Heterogeneous fleet, Mixed Backhauls, and Time Windows.



To our knowledge, the proposed algorithm based on ACO algorithm coupled with local search algorithms depicted in this paper is the first effective constructive algorithm for the addressed problem in this study.

The remainder of the paper is organized as follow. Section 2 proposes a mixed integer linear program for the PDPTW problem. Section 3 describes the ACO, with the pheromone initialization, their updating and the computation of the visibility. Section 4 presents three local search algorithms. The setting of parameters and the experimental results are reported in Sect. 5.

## 2 Mathematical Model

This section presents a new mixed integer linear program to model the addressed problem. It is based on the model proposed by [11] for the PDPTW with heterogeneous fleet. We note  $N$  the set of  $2n$  customers, node 0 represents depot (origin and destination) and  $V = N \cup \{0\}$  the set of  $2n + 1$  nodes.  $P = \{1, \dots, n\}$  is the set of pickup demands, the set of  $n$  delivery demands is noted  $D = \{n + 1, \dots, 2n\}$ ,  $K$  is the set of  $m$  identical vehicles with a capacity of  $Q$  items.  $A$  is the set of arcs,  $\delta$  is the fixed cost of using a vehicle,  $d_{ij}$  represents the distance between the vertices  $i$  and  $j$ ,  $t_{ij}$  is the time between the location of vertices  $i$  and  $j$ ,  $s_i$  represents the service time required by the node  $i$ ,  $|q_i|$  is the amount of goods to pickup or delivery,  $e_i$  the earlier time at which the service may begin at node  $i$  and  $l_i$  the latest time at which the service may begin at node  $i$ . We assume that:

$$\forall i \in P, q_i > 0 \text{ and } q_{n+i} = -q_i \text{ and } (i,j) \in A \iff e_i + s_i + t_{ij} \leq l_j.$$

The problem is formulated as a mixed integer linear program (MILP). For each arc  $(i,j) \in A$  and each vehicle  $k \in K$ , let  $x_{ij}^k$  be a binary variable equals to 1 if the vehicle  $k$  travels from location of demand  $i$  to location of demand  $j$ , and 0 otherwise. For each node  $i \in N$ , and each vehicle  $k \in K$ , let  $B_i^k$  the time at which vehicle  $k$  begins the service at node  $i$ .  $Q_{ij}^k$  is the load of vehicle  $k$  on the arc  $(i,j)$ . The formulation is the following:

$$Min \quad \sum_{i \in P} \delta x_{0i}^k + \sum_{k \in K} \sum_{i \in N} \sum_{j \in N} d_{ij} x_{ij}^k \tag{1}$$

$$\sum_{(i,j) \in A} \sum_{k \in K} x_{ij}^k = 1, \quad \forall i \in P; \tag{2}$$

$$\sum_{j|(i,j) \in A} x_{ij}^k = \sum_{j|(n+i,j) \in A} x_{n+i,j}^k, \quad \forall (i,k) \in P \times K; \tag{3}$$

$$\sum_{i \in P} x_{0i}^k \leq 1, \quad \forall k \in K; \tag{4}$$

$$\sum_{i \in P} x_{0i}^k = \sum_{i \in D} x_{i0}^k, \quad \forall k \in K; \quad (5)$$

$$\sum_{j|(i,j) \in A} x_{ij}^k = \sum_{j|(j,i) \in A} x_{ji}^k, \quad \forall (i, k) \in P \cup D \times K; \quad (6)$$

$$\sum_{j|(i,j) \in A} x_{ij}^k = \sum_{i \in P} x_{0i}^k, \quad \forall (i, k) \in P \cup D \times K; \quad (7)$$

$$\sum_{j|(i,j) \in A} \sum_{k \in K} Q_{ij}^k + \sum_{j|(j,i) \in A} \sum_{k \in K} Q_{ji}^k = q_i, \quad \forall i \in P; \quad (8)$$

$$\sum_{i \in P} \sum_{k \in K} Q_{0i}^k + \sum_{i \in D} \sum_{k \in K} Q_{i0}^k = 0; \quad (9)$$

$$Q_{ij}^k \leq Q \times x_{ij}^k, \quad \forall (i, j) \in A, \forall k \in K; \quad (10)$$

$$B_i^k - l_i + (l_i + s_i + t_{ij})x_{ij}^k \leq B_j^k, \quad \forall (i, j) \in A, \forall k \in K; \quad (11)$$

$$e_i \sum_{j|(i,j) \in A} x_{ij}^k \leq B_i^k \leq l_i \sum_{j|(i,j) \in A} x_{ij}^k, \quad \forall (i, k) \in N \times K; \quad (12)$$

$$B_i^k + (s_i + t_{i,n+i}) \times \sum_{j \in N} x_{ij}^k \leq B_{n+i}^k, \quad \forall (i, k) \in P \times K; \quad (13)$$

$$x_{ij}^k \in \{0, 1\}, \quad Q_{ij}^k \geq 0, \quad B_i^k \geq 0, \quad \forall (i, j) \in A, \forall k \in K. \quad (14)$$

The objective function (1) minimizes the number of vehicles used and the total distance travelled. Constraints (2) and (3) ensure that each pickup demand is served exactly once and the corresponding delivery demand is served by the same vehicle in the same route. Constraints (4) and (7) guarantee that the route of each vehicle starts and ends at the depot. The respect of the time windows and the capacity of vehicle is ensured by constraints (8) to (12). Constraint (13) assures that every delivery demand is satisfied after the corresponding pickup demand but not necessary immediately after the pickup point.

### 3 Ant Colony Optimization (ACO)

In this study, we apply a variant of ACO called Ant Colony System (ACS), characterized by introduction of a local pheromone update. Ant colony optimization is chosen because it is a constructive method which does not require reparation procedure.

### 3.1 Construction of Solution

A solution is composed of a set of routes and each route is realized by one vehicle. An ant constructs the routes of a solution sequentially. Ant keeps inserting demands in the current route as long as the are non-satisfied demands that respect the constraints (capacity, times windows and paring). If in a partial solution there still are non-visited nodes but none of them can be inserted in the route being built, the ant close the current route and start a new one. Let  $\rho$  a partial solution formed by  $r - 1$  complete routes and a  $r^{th}$  route in construction. Let  $i$  the last demand completed in route  $r$ ,  $Q_r$  the load of the vehicle after satisfied demand  $i$  and  $\mathcal{V}$  the set of unsatisfied delivery demands such that their corresponding pickup demands has been served. A demand  $j$  is *eligible* to be satisfy if it didn't have been completed yet and if one of these conditions is satisfied:

1.  $0 < j \leq n$  and there exists a path to satisfied all demands in  $\mathcal{V} \cup \{n + j\}$
2.  $n < j \leq 2n \wedge j - n \in r$  and there exists a path to satisfied all demands in  $\mathcal{V} \setminus \{j\}$

An eligible demand  $j$  to be inserted in the partial solution  $\rho$  is chosen randomly using probability:

$$P_{ij}^\rho = \frac{(\tau_{ij})^\alpha (\eta_{ij}^\rho)^\beta}{\sum_{d \in S_i^\rho} (\tau_{id})^\alpha (\eta_{id}^\rho)^\beta} \text{ if } j \in S_i^\rho, \text{ and } 0 \text{ otherwise.} \tag{15}$$

$P_{ij}^\rho$  represents the probability to choose a demand  $j$  to complete from the current demand  $i$ .  $\tau_{ij}$  denotes the trail of pheromone on arc  $(i, j)$ .  $S_i^\rho$  is the set of eligible demands that we can performed after demand  $i$ . The parameters  $\alpha$  et  $\beta$  modulate the importance between the visibility and the pheromone.  $\eta_{ij}^\rho$  is the visibility value used to guide ant.

$$\eta_{ij}^\rho = \frac{\alpha_1 |S_j^{\rho \cup \{i\}}|}{d_{ij}} \tag{16}$$

with  $\alpha_1 \in ]0, 1]$  a fixed scalar.

Given a partial solution  $\rho$ , ending by satisfying demand  $i$ , the eligibility of a demand  $j$  is obtained by finding a feasible path in a graph. Indeed, it shall be demonstrate that after performed demand  $j$ , there exists a feasible path to satisfy all unsatisfied delivery demands whose corresponding pickup demands were satisfied in route being built in  $\rho$ . This problem is a Hamiltonian path problem, which is NP-complete and to solve it, an insertion heuristic Algorithm 1 is proposed.

### 3.2 Pheromone Updating

At the beginning of ACS, pheromones are initialized by:

---

**Algorithm 1** Insertion heuristic for Hamiltonian path problem
 

---

```

1: Inputs:  $\mathcal{V}$  (a set of demand to satisfy),  $ItMax1$  (a number of iterations),  $i$  (the current demand)
   and  $t$  (the time at the end of service of demand  $i$ )
2: while number of iterations  $< ItMax1$  and no feasible path which satisfy all demands in  $\mathcal{V}$  is
   found do
3:    $\mathcal{V}' = \mathcal{V}$ 
4:   Initialize a route  $r$  beginning at node  $i$  at the time  $t$ 
5:   while  $\mathcal{V}'$  in not empty do
6:     Choose a random delivery  $d \in \mathcal{V}'$ , remove  $d$  in  $\mathcal{V}'$ , and try to insert it in  $r$  at the best
     position.
7:     if  $d$  is not inserted in  $r$  then
8:       Go back to step 3.
9:     end if
10:  end while
11: end while

```

---

$$\tau_{ij} = \tau_0 \text{ if } (i, j) \in A, \text{ and } 0 \text{ otherwise.} \quad (17)$$

with  $\tau_0$  a fixed scalar. As we mentioned, ACS has two types of pheromone update:

- local updating :  $\tau_{ij} = \epsilon_1 \tau_{ij} + \tau_0$ ,

used to diversify the search in a given iteration.

- global updating used:

$$\tau_{ij} = \epsilon_2 \tau_{ij} + (1 - \epsilon_2) \Delta_{ij}^* \text{ if } (i, j) \text{ belong in the best ant, and } \epsilon_2 \tau_{ij} \text{ otherwise.} \quad (18)$$

with  $\Delta_{ij}^* = \frac{\text{number of demands in } r_i^{*a_2}}{\text{total distance travelled on } r_i^*}$ ,  $r_i^*$  the route which contain the demand  $i$ ,  $\epsilon_1, \epsilon_2 \in ]0, 1[$  fixed and  $\alpha_2$  a positive fixed scalar.

## 4 Local Search Algorithm

This section describes three local search algorithms used in this work. Each local search is dedicated to specific feature of the objective function of PDPTW.

Heuristic **H1** is inspired from the two-stage method used in [6]. For a given solution, H1 is used to decrease the number of vehicles. For a each route in the solution, H1 removes a route from it, and try to insert all its requests in the remaining routes of the solution. The order of insertion is the order of completion in the delete route. Every request is inserted in the route and in the positions (pickup and delivery positions) that minimize the total distance travelled. If finally, all the request presents in the deleted route are inserted, the solution is updated.

The second heuristic **H2** is proposed to reduce distance travelled for a given route. The idea is to generate randomly (uniform distribution) an insertion order of pickup demand. And inserted at the best position the requests in this order.

The third local search **H3** is proposed to minimize total distance travelled for a given solution. At each iteration, a route  $r$  and a demand  $d$  (in  $r$ ) are chosen randomly. The corresponding couple of pickup and delivery demands for  $d$  is remove from the solution and reinserted in the route and at the position whom minimise total cost. H3 is inspired from a part of LNS algorithm developed by S. Ropke in [8]. The pseudo code of hybrid ACS used is given by Algorithm 2.

---

### Algorithm 2 Pseudo code of hybrid ACS

---

```

Initialization of parameters
while the best solution is not improved in ItMax iterations do
  for each ant of the population do
    Construct a solution to complete all demands
    while we can remove a route do
      Apply respectively algorithms: H2, H1 and H3
    end while
    if the current solution have the same number of vehicles as the best solution found then
      while we decrease the total distance travelled do
        Apply respectively algorithms: H2 and H3
      end while
    end if
    Apply the local updating pheromone
  end for
  Apply the global updating pheromone
end while

```

---

In the algorithm, the order H2, H1 and then H3 is used to first optimize each route, and then try to decrease the vehicles number and finally, minimize the total distance travelled.

## 5 Computational Results

The proposed approach has been implemented on eclipse, the programming language was C++ and the experiments have been carried on a 1.5 GHz and 3.3 GB of RAM. Standard Li et Lim's benchmark [9] is chosen to evaluate the performance of our approach. For experiments, we keep the same value  $\epsilon = 0.9$  proposed by Belmecheri et al. [2]. A sensibility analysis is made in order to fix the following parameters: Number of ants  $\in \{\frac{n}{2}, \frac{2n}{5}, \frac{n}{3}\}$ ,  $\alpha \in \{2, 3, 4\}$ ,  $\beta \in \{1, 2\}$ ,  $\alpha_1 \in \{0.1, 0.2, 0.5, 1\}$ ,  $\alpha_2 \in \{2, 3, 5, 10\}$ ,  $\tau_0 \in \{0.01, 1, 10\}$ ,  $ItMax1 \in \{5, 10, 20\}$  and  $Itmax2 \in \{n, 2n, 3n\}$ . We obtain with this analysis that the best values are : Number of ants =  $\frac{2n}{5}$ ,  $\alpha = 3$ ,  $\beta = 1$ ,  $\alpha_1 = 0.2$ ,  $\alpha_2 = 5$ ,  $\tau_0 = 1$ ,  $ItMax1 = 10$ ,  $Itmax2 = 2n$ . The algorithm stops when the best solution is not improved after 100 iterations.

**Table 1** Experiments of ACO algorithm

Instance	Best known solution				Hybrid ACS				Best known solution				Hybrid ACS					
	NV	TD	REF	NV	TD	CPU	NV	REF	NV	TD	CPU	Instance	NV	TD	REF	NV	TD	CPU
lc101*	10	828,94	[11]	10	828,94	86	10		9	1003,77		lr112	9	1003,77	[5]	9	<b>1002,38</b>	167
lc102	10	828,94	[5]	10	828,94	103	10		4	1253,23		lr201	4	1253,23	[9]	4	1253,23	207
lc103	9	1035,35	[2]	<b>9</b>	<b>968,92</b>	96	<b>9</b>		3	1197,67		lr202	3	1197,67	[5]	3	1197,67	455
lc104	9	860,01	[9]	9	860,01	156	9		3	949,4		lr203	3	949,4	[5]	3	949,4	556
lc105	10	828,94	[5]	10	828,94	37	10		3	849,05		lr204	3	849,05	[5]	<b>3</b>	<b>847,83</b>	906
lc106	10	828,94	[5]	10	828,94	43	10		3	1054,02		lr205	3	1054,02	[5]	<b>3</b>	<b>1053,98</b>	134
lc107	10	828,94	[5]	10	828,94	39	10		3	931,63		lr206	3	931,63	[5]	3	931,63	641
lc108	10	826,44	[5]	10	826,44	49	10		2	903,06		lr207	2	903,06	[5]	<b>2</b>	<b>902,24</b>	434
lc109	9	1000,60	[2]	10	827,82	96	10		2	734,85		lr208	2	734,85	[5]	<b>2</b>	<b>734,09</b>	1555
lc201*	3	591,56	[11]	3	591,56	49	3		3	930,59		lr209	3	930,59	[9]	3	930,59	234
lc202*	3	591,56	[11]	3	591,56	260	3		3	964,22		lr210	3	964,22	[5]	3	964,22	375
lc203	3	591,17	[9]	3	591,17	190	3		2	911,52		lr211	2	911,52	[9]	<b>2</b>	<b>903,76</b>	1525
lc204	3	590,6	[9]	<b>3</b>	<b>590,39</b>	634	<b>3</b>		14	1708,8		lr101	14	1708,8	[5]	14	1708,77	98

(continued)

**Table 1** (continued)

	Best known solution				Hybrid ACS				Best known solution				Hybrid ACS			
lc205*	3	588,88	[11]	3	588,88	77		irc102	12	1558,07	[9]	12	<b>1556,82</b>	49		
lc206	3	588,49	[5]	3	588,49	144		irc103	11	1258,74	[5]	11	<b>1256,06</b>	57		
lc207	3	588,29	[5]	3	588,29	102		irc104	10	1128,40	[5]	10	<b>1126,23</b>	98		
lc208	3	588,32	[5]	3	588,32	108		irc105	13	1637,62	[5]	13	<b>1633,56</b>	52		
lr101	19	1650,80	[5]	19	1650,8	31		irc106	11	1424,73	[9]	11	1424,73	210		
lr102	17	1487,57	[5]	17	<b>1487,49</b>	54		irc107	11	1230,14	[9]	11	<b>1225,68</b>	58		
lr103	13	1292,68	[5]	13	<b>1284,93</b>	61		irc108	10	1147,43	[9]	10	1147,96	70		
lr104	9	1013,39	[5]	9	<b>999,27</b>	87		irc201	4	1406,94	[9]	4	<b>1396,88</b>	195		
lr105	14	1377,11	[5]	14	1377,11	31		irc202	3	1374,27	[5]	3	<b>1361,24</b>	153		
lr106	12	1252,62	[5]	12	<b>1248,93</b>	37		irc203	3	1089,07	[5]	3	1089,07	350		
lr107	10	1111,31	[5]	10	<b>1101,89</b>	59		irc204	3	818,66	[9]	3	818,66	885		
lr108	9	968,97	[5]	9	<b>966,30</b>	59		irc205	4	1302,20	[5]	4	1302,20	274		
lr109	11	1208,96	[9]	11	<b>1208,92</b>	43		irc206	3	1159,03	[9]	3	<b>1156,54</b>	135		
lr110	10	1159,35	[5]	10	<b>1158,27</b>	113		irc207	3	1062,05	[9]	3	<b>1054,24</b>	205		
lr111	10	1108,9	[5]	10	1108,9	74		irc208	3	852,76	[5]	3	852,76	469		

Table 1 contains five types of columns: *Instance* is the name of instance, *NV* is the number of used vehicles, *TD* is the total travelled distance, *REF* is a reference to the paper which found the result and *CPU* is the computational time in seconds. The symbol “\*” indicates that the instance is solved to optimality and the values in bold emphasize that the proposed algorithm found a better solution.

The hybrid ACS algorithm returns in 98.2% (55/56) of cases a solution better or equal to the best known solution. And find a better solution than the best know solution in 44.6% (25/56). It is important to remark that our method is able to performed best known solutions on all configurations: lc (clustered), lr (Uniform distributed) and lrc (Semi-clustered).

## 6 Conclusion

This paper proposes an efficient algorithm to solve a Pickup and Delivery Problem with Time Windows with objective function : first minimize the number of vehicles and second minimize the total distance travelled. The proposes algorithm is based on ant colony optimization coupled with three fast local search algorithms. To our knowledge, this approach is the first constructive method to solve PDPTW problem with this objective. The experiments on the standard PDPTW benchmark of Li and Lim [10] show that it outperforms existing algorithms. In the future, it would be interesting to performed every feature of approach (construction, visibility, update of pheromone and local search algorithms) to solve more large size instances and generalized this approach on heterogeneous fleet case.

## References

1. Belmecheri, F., et al.: An ant colony optimization algorithm for a vehicle routing problem with heterogeneous fleet, mixed backhauls, and time windows. In: 13th IFAC Symposium on Information Control Problems in Manufacturing. Moscow, Russia (2009)
2. Bent R, van Hentenryck P.: A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows. In: Rossi, F. (ed.) Principles and Practice of Constraints Programming, Springer, Heidelberg, Berlin (2003)
3. Dumas, Y., Desrosiers, J.: The pickup and delivery problem with time windows. Eur. J. Operation. Res. **54**, 7–22 (1991)
4. Goss, S., Aron, S., Deneubourg, J., Pasteels, J.: Self-organized shortcuts in the Argentine ant. Naturwissenschaften **76**, 579–581 (1989)
5. Li, A., Lim, H.: A metaheuristic for the pickup and delivery vehicle routing problems with time windows. In: IEEE Computer Society, 13th IEE International Conference on Tools with Artificial Intelligence (ICTAI-01), pp. 333–340. Los Alamitos, USA (2001)
6. Nagata, Y., Kobayashi.: Guided ejection Search for the pickup and delivery problem with time windows. In: Cowling, P., Merz, P. (eds.) Evolutionary Computation in Combinatorial Optimization, Springer, Berlin, Heidelberg (2010)



7. Nalepa, J., Blocho, M.: A parallel algorithm with the search space partition for the pickup and delivery with time windows. In: Proceedings of 10th International Conference on P2P, Parallel, Grid, Cloud and internet Computing (IEE 3PGCIC), pp. 92–99 (2015)
8. Parragh, S., Doerner, K., Hartl, R.: A survey on pickup and delivery problems. *J. fur Betriebswirtschaft* **58**(2), 81–117 (2008)
9. Ropke, S., Pisinger, D.: An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Trans. Sci.* **40**(4), 455–472 (2006)
10. SINTEF vehicle routing and traveling salesperson problems. <https://www.sintef.no/projectweb/top/pdptw/li-lim-benchmrak/100-customers/>
11. Tchoupo, M.N., Yalaoui, A., Amodeo, L., Yaloui, F., Lutz, F.: Problème de collectes et livraisons avec fenêtres de temps et flotte hétérogène. In: 11th International Conference on Modeling, Optimization & Simulation. Montreal, Canada (2016)

**Part V**  
**Innovative Applications**

# Initialization of Optimization Methods in Parameter Tuning for Computer Vision Algorithms

Andrea Bessi, Daniele Vigo, Vincenzo Boffa and Fabio Regoli

**Abstract** Computer Vision Algorithms (CVA) are widely used in several applications ranging from security to industrial processes monitoring. In recent years, an interesting emerging application of CVAs is related to the automatic defect detection in some production processes for which quality control is typically performed manually, thus increasing speed and reducing the risk for the operators. The main drawback of using CVAs is represented by their dependence on numerous parameters, making the tuning to obtain the best performance of the CVAs a difficult and extremely time-consuming activity. In addition, the performance evaluation of a specific parameter setting is obtained through the application of the CVA to a test set of images thus requiring a long computing time. Therefore, the problem falls into the category of expensive Black-Box functions optimization. We describe a simple approximate optimization approach to define the best parameter setting for a CVA used to determine defects in a real-life industrial process. The algorithm computationally proved to obtain good selections of parameters in relatively short computing times when compared to the manually determined parameter values.

---

A. Bessi · D. Vigo (✉)  
DEI, University of Bologna, Bologna, Italy  
e-mail: daniele.vigo@unibo.it

A. Bessi  
e-mail: andrea.bessi3@unibo.it

V. Boffa · F. Regoli  
Pirelli Tyre S.p.A, Milano, Italy  
e-mail: vincenzo.boffa@pirelli.com

F. Regoli  
e-mail: fabio.regoli@pirelli.com

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_20

## 1 Introduction

The calibration of computer vision algorithms (CVAs) is a time consuming and critical step in the effective use of CVAs in many applications, such as the automated defect detection of pieces produced by an industrial plant. In this case, the final quality control check at the end of the production chain consists in the optical scan of the produced pieces by a set of several CVAs. Each of them is designed to detect a specific type of defect and its behavior is controlled by a large set of parameters, which influence the CVA sensibility and accuracy and must be determined to maximize its detection efficacy on specific types of images. For a general overview of automated defect detection see [7] (see also [5] for an example in the textile industry).

More precisely, given a set of images, the efficacy of the error detection is measured as a function of the positive and negative false ratios produced by the CVA with a specific parameter set. As in many other applications parameter tuning of the CVAs is, therefore, a crucial component for the overall efficacy of the system. To the best of our knowledge, no optimization method has been developed so far for parameter tuning in defect detection.

In the context of CVAs, the computation of the efficacy requires the application of the CVA to a training set of test images. This is typically a very time-consuming operation requiring several seconds per image, hence minutes or even hours for a significant training set. Therefore, approaches based on black-box function optimization (see, e.g., [2, 3]) must be used in this case. To this end, we developed a simple Sequential Approximate Optimization (SAO) algorithm (see, e.g., [4]) to identify the optimal parameter values for a CVA used to detect a specific error on the images. During the optimization process, the solutions iteratively found by the algorithm are evaluated by executing the target CVA on the training set of images. The comparison between the CVA outputs obtained on the images and their real defectiveness state produce the true/false positive index ratio for the solutions tested. Our goal is the determination of the optimal input parameter combination for the CVA, leading to the best possible false positive and negative ratios for each particular type of defect.

In Sect. 2 we describe in detail the characteristic of the problem under study. In Sect. 3 the structure of the proposed algorithm is given and in Sect. 4 we present the results of an experimental validation of the algorithm on data coming from a specific real-world application.

## 2 Problem Definition

The calibration of the parameters of a CVA is an optimization problem which can be described as follows. The variables to be optimized are the input parameters of the CVA which are assumed here to be continuous and associated with a lower and an upper bound) for their variation. The performance of the CVA is measured in terms of two independent indicators, namely the number of false-negative and false-positive in the solution, to be defined later.

More precisely, we have a CVA whose behavior depends on a subset  $I$  of parameters whose value has to be determined. For each parameter  $i \in I$  we are given a lower and upper bounds  $l_i$  and  $u_i$ , respectively. For each parameter  $i \in I$  let  $x_i$  be the decision variable which represents its value. Given solution  $\vec{x}$  we can evaluate its quality by measuring the performance of the CVA on a training set  $S$  of images. To this end, let  $f_p(\vec{x})$  be the number of false-positives returned by the CVA when applied to the set  $S$  with parameters  $\vec{x}$ , defined as the number of non-defective images which are classified as defective by the CVA. Similarly, let  $f_n(\vec{x})$  be the number of false-negatives returned by the CVA, defined as the number of defective images which are classified as non-defective by the CVA. Finally, let  $\alpha_p$  and  $\alpha_n$  be two nonnegative weights associated with the two performance measures. The CVA Parameter Tuning Problem (CVAPTP) can be formulated as follows

$$(CVAPTP) \quad z = \min F(\vec{x}) = \{\alpha_p f_p(\vec{x}) + \alpha_n f_n(\vec{x})\}, \quad s.t. \quad l_i \leq x_i \leq u_i \quad \forall i \in I. \quad (1)$$

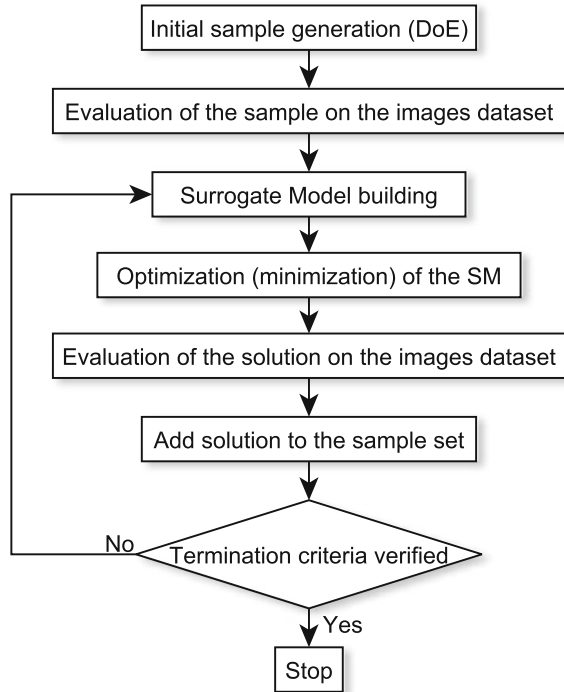
### 3 A Sequential Approximate Optimization Algorithm

The Black Box Optimization (BBO) nature of the problem requires the use of an inference intelligence able to predict the objective function value of an unsampled solution to guide the search process. In the literature, such representation of the BB function it is referred to as the Surrogate Model (SM, see [8]) for which a large number of types and formulations were proposed.

As frequently done in the recent literature the SM is used within a Sequential Approximate Optimization (SAO) algorithm that iteratively updates the SM by adding the solution points that are determined at each iteration. This sequential approach preserves a certain simplicity but provides some important advantages. First, the rebuilding of the SM in order to capture the incoming information improves its reliability at each iteration. Second, it guarantees to perform a global optimization over the entire solutions domain, by reducing the possibility to being trapped into local optima. The general scheme of the simple SAO algorithm we adopted is depicted in Fig. 1.

The algorithm starts with the identification of the initial sample of solution points, used to initialize the SM. Then, for each sample point, the corresponding value of the objective function  $F(\vec{x})$  in (1) is computed by applying the parameter values associated with the point to the CVA over the entire images test set. The incumbent solution is defined as the best solution found so far, hence it initially corresponds to the best sample. The SM is built from the current set of points  $(\vec{x}, F(\vec{x}))$  and used to determine the next candidate solution. After this, the SM is interrogated in order to search for the best candidate solution possibly improving the incumbent. This phase is generally called *adaptive sampling* criteria. The process is iterated until termination criteria based on solution quality and running time are met.

**Fig. 1** Outline of the sequential approximate optimization algorithm



As to the type of SM used, in this work due to the BBO nature of the problem we adopted a specific type of the so called Meshfree methods, named Radial Basis Function (RBF) interpolation techniques. These are relatively easy to construct and are widely used to approximate Black Box function responses. In RBF interpolation the model,  $s(\vec{x})$ , is defined as the sum of a given number  $K$  of radial functions  $\phi$ :

$$s(\vec{x}) = \sum_{j=1}^K \gamma_j \phi(\|\vec{x} - \vec{x}_j\|) \quad (2)$$

where  $\vec{x}_j, j = 1, \dots, K$ , is the set of sampled solution points representing the centers of the radial functions  $\phi()$ ,  $\gamma$  is a vector of weights to be determined, and  $\vec{x}$  is the unsampled point whose value has to be predicted. Regarding the radial functions  $\phi$ , several type are available in literature, varying from parametrized to not parametrized ones. In our case, the best trade off between a simple construction of the SM and an acceptable reliability resulted with the use cubic basis function, that assume the form:

$$\phi(\|\vec{x} - \vec{x}_j\|) = \|\vec{x} - \vec{x}_j\|^3 \quad (3)$$

We refer the reader to [1] for an overview of Meshfree methods, and to [6] for an example of industrial use of cubic RBF.

As to the initial sample generation through which initialize the SM, several techniques exists in literature, typically referred to as Design of Experiment (DoE). Since in our problem the parameters  $x_j$  are not subject to other constraints besides the upper and lower bounds, classical DoE as the Factorial Design are suitable. In particular, a Full Factorial Design (FFD) permit to cover the entire domain space, selecting all the points of the grid generated by the discretization of each design variable (i.e., the parameters in our problem). This methodology is appropriate with the cubic RBF interpolation since it guarantees a sufficient reliability only inside the convex hull of the sampled points. However, in our case the computation of  $F(\vec{x})$  is extremely time-consuming and using grids in which the parameter's values are discretized is not practically possible. For this reason, we decided to initialize our algorithm through a  $|I|^2$  FFD, using just the domain vertices obtained with the lower and upper bounds of the parameters to be optimized. To improve the initial sample quality we also considered an initialization in which  $H$  additional random points selected inside the domain hypercube are considered. The adaptive sampling strategy that we adopt to perform the search of the candidate solution on the SM is the minimization of its predictor  $s(\vec{x})$ . To avoid to being trapped in a local minimum and perform an efficient search over all the domain, we implement a multi-start gradient descent algorithm and run it by using a discrete grid of starting points.

Finally, we terminate the algorithm after a maximum number of objective function evaluations or after a given number of non-improving iterations.

## 4 Experimental Validation

We applied our algorithm to the tuning of a CVA used to detect a specific type of defects on tyre images obtained in a real-world production environment. The training set is made up of 160 images for which the presence or absence of defects is known. Six parameters were selected as the target for the optimization. Each such parameter has a maximum and minimum value and a default value manually determined by the CVA designers. The behavior of the CVA with the default parameter values is used here as a benchmark reference to evaluate the performance of the optimized parameters set.

We tested the impact of three variants for the initialization step, leading to three different overall algorithms  $A_1, A_2$ , and  $A_3$ . In  $A_1$  we used  $H_1 = 100$  random points to initialize the algorithm. In  $A_2$  the sample set is constituted by the  $2^6$  points of the simple FFD described in Sect. 3. Finally, in  $A_3$ , we added to the FFD set  $H_3 = 36$  random internal points. The overall algorithm is run for a total of 200 objective function evaluations (including those for the initialization step). The gradient descent search for the candidate solution is performed from a  $9^6$  discrete grid of points.

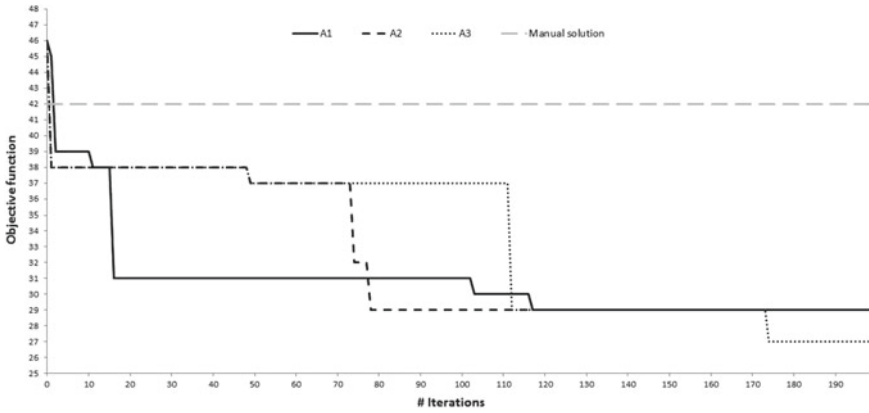


Fig. 2 Evolution of the proposed algorithms with weight combination (1,5) in comparison with the manual tuning

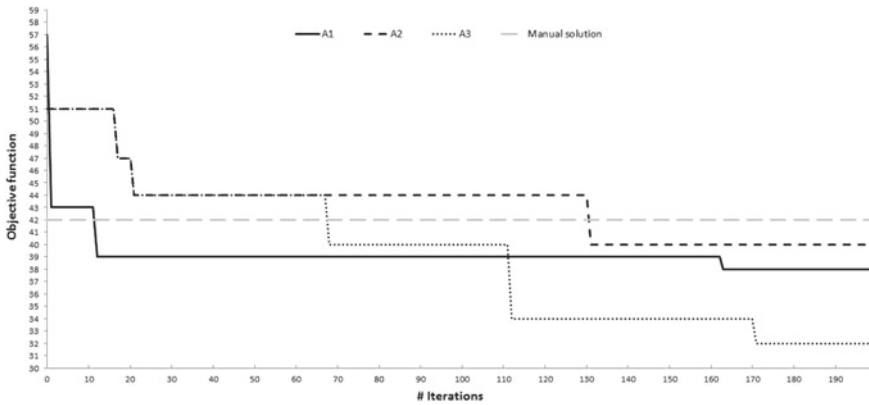


Fig. 3 Evolution of the proposed algorithms with weight combination (1,10) in comparison with the manual tuning

To account for possible different relative importance of false positives and false negatives in the defect detection, we considered two different pairs of weights in the objective function (1). Namely, we considered  $(\alpha_p, \alpha_n) = \{(1, 5), (1, 10)\}$ .

The results for the three algorithms are illustrated in Figs. 2 and 3 for the (1,5) and (1,10) weight combinations, respectively. The figures report the evolution of the objective function for each algorithm compared with the benchmark reference equal to 42 for both weight combinations. By observing the figures it clearly appears that all proposed algorithms generate better parameter combinations with respect to the manual ones. In particular, for the (1,5) case  $A_1, A_2$  and  $A_3$  produce solutions with value 29, 29 and 27, respectively, which are 31% and 36% better than manual ones. For the (1,10) case, they find a solution with value 38, 40 and 32, which are



10%, 5%, and 24% better, respectively. In general, we can observe that the mixed initialization of  $A_3$  provides better final results but the simple FFD of  $A_2$  improves quite rapidly and may constitute a good alternative when less time is available.

## References

1. Fasshauer, G.: *Meshfree Approximation Methods with Matlab*. World Scientific Publishing Co., Inc., River Edge, NJ, USA (2007)
2. Jones, D.: A taxonomy of global optimization methods based on response surfaces. *J. Global Optim.* **21**, 345–383 (2001)
3. Jones, D., Schonlau, M., Welch, W.: Efficient global optimization of expensive black-box functions. *J. Global Optim.* **13**, 455–492 (1998)
4. Kitayama, S., Arakawa, M., Yamazaki, K.: Sequential approximate optimization using radial basis function network for engineering optimization. *Optim. Eng.* **12**, 535–557 (2011)
5. Kumar, A.: Computer-vision-based fabric defect detection: a survey. *IEEE Trans. Indust. Elect.* **55**(1), 348–363 (2008)
6. McDonald, D., Grantham, W., Tabor, W., Murphy, M.: Global and local optimization using radial basis function response surface models. *Appl. Mathemat. Model.* **31**, 2095–2110 (2007)
7. Newman, T., Jain, A.: A survey of automated visual inspection. *Comput. Vis. Image Understand.* **61**(2), 231–262 (1995)
8. Queipo, N., Haftka, R., Shyy, W., Goel, T., Vaidyanathan, R., Tucker, P.: Surrogate-based analysis and optimization. *Progr. Aerosp. Sci.* **41**, 1–28 (2005)

# Using OR + AI to Predict the Optimal Production of Offshore Wind Parks: A Preliminary Study

Martina Fischetti and Marco Fraccaro

**Abstract** In this paper we propose a new use of Machine Learning together with Mathematical Optimization. We investigate the question of whether a machine, trained on a large number of optimized solutions, can accurately estimate the value of the optimized solution for new instances. We focus on instances of a specific problem, namely, the offshore wind farm layout optimization problem. In this problem an offshore site is given, together with the wind statistics and the characteristics of the turbines that need to be built. The optimization wants to determine the optimal allocation of turbines to maximize the park power production, taking the mutual interference between turbines into account. Mixed Integer Programming models and other state-of-the-art optimization techniques, have been developed to solve this problem. Starting with a dataset of 2000+ optimized layouts found by the optimizer, we used supervised learning to estimate the production of new wind parks. Our results show that Machine Learning is able to well estimate the optimal value of offshore wind farm layout problems.

**Keywords** Machine learning · Mixed integer linear programming · Wind energy

---

M. Fischetti (✉)

Vattenfall and Technical University of Denmark, DTU Management Engineering,  
Produktionstorvet 424, 2800 Kgs. Lyngby, Denmark  
e-mail: martfi@dtu.dk

M. Fraccaro

Technical University of Denmark, DTU Compute,  
Richard Petersens Plads 321,  
2800 Kgs. Lyngby, Denmark  
e-mail: marfra@dtu.dk

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_21

## 1 Introduction

Mathematical Optimization (MO) and Machine Learning (ML) are two very popular disciplines. A successful application of the two together arises in the so-called *Prescriptive Analytics* field [2], where ML is used to predict a phenomenon, and MO techniques are used to optimize an objective over that prediction. In the present work, we will instead investigate a different way to merge MO and ML, where the optimization model comes first, and its optimized solutions are used as training set for Linear Regression and Neural Networks. The idea can be used for many different optimization problems and applications (transport, logistic, scheduling, etc.). We will focus on a specific application, already studied by the first author in [5], namely the offshore wind park layout optimization problem.

The *wind farm layout optimization problem* consists of finding an optimal allocation of turbines in a given offshore site, to maximize the park power production. A particularly challenging feature of this problem is the interaction between turbines, also known as *wake effect*. The wake effect is the interference phenomenon for which, if two turbines are located close to each other, the upwind one creates a shadow on the one behind. This is of great importance in the design of the layout since it results into a loss of power production for the turbine downstream, that is also subject to a possibly strong turbulence. For many years, this problem has been unknown or underestimated, and old wind parks have been designed with a very regular (and highly wake-affected) layout. It was estimated in [1] that, for large offshore wind farms, the average power loss due to turbine wakes is around 10–20% of the total energy production. It is then obvious that power production can increase significantly if the wind farm layout is properly optimized. However, the large size of the problem, the complexity of the wake effect, and the presence of other constraints, makes it impossible to create a good layout without the usage of an advanced optimization tool. Since the difference in power production between optimized solution and unoptimized ones can be significant, it is even difficult to estimate the potential power production of a site, without running a complete optimization of the layout. In this paper, we aim at developing a ML algorithm able to better estimate the potential of a site, without running a complete optimization. Such a ML algorithm could be used, for example, in an early stage of the project, when the company has to decide where to build the park. Having, for example, the possibility of selecting among a (possibly) large number of offshore sites, the ML algorithm could quickly estimate which of the sites has the highest expected power. Once the site is selected, a detailed (and more time-consuming) optimization can be run to define the actual layout. To be more specific, in the present paper we address the case where a company wants to construct a specific number of turbines in an offshore area. Even if the production would increase by spreading the turbines to reduce the wake effect, the infrastructure costs to connect turbines very far away would also increase. Therefore we assume that, even if a large sea area is available, the company would discretize it in a number of smaller rectangular sites. These sites will typically have different dimensions, and the wind can greatly vary from site to site. The company could also be inter-

ested in investigating the potential of different sites for different turbine types. We therefore considered rectangular instances of different dimensions, with different wind scenarios (taken from real-world parks) and with different turbine types. We defined and optimized over 2000 instances using the MO tool developed in [5]. The power production of all these optimized instances is used as training set for our ML algorithm.

A distinctive feature of our work is that we do not expect to estimate the optimal *solution* (which is arguably very problematic for the ML state of the art), but we content ourselves with the estimate of the optimal *value* of it.

## 2 The Optimization Model

At the optimization stage, an offshore site is given together with the wind statistics in the site. We are asked to determine the optimal allocation of turbines in the area, in order to maximize the park power production. The optimization needs to consider that a minimum and maximum number of turbines can be built, a minimal separation distance must be guaranteed between two turbines to ensure that the blades do not physically clash (turbine distance constraints), and the power loss due to wake effect.

We first discretize the area in a number of possible turbine positions. Let  $V$  denote this set and let

- $I_{ij}$  be the interference (loss of power) experienced at position  $j$  when a turbine is installed at position  $i$ , with  $I_{jj} = 0$  for all  $j \in V$ ;
- $P_i$  be the power that a turbine would produce if built (alone) at position  $i$ . We used a Jensen's model to compute it [6];
- $N_{MIN}$  and  $N_{MAX}$  be the minimum and maximum number of turbines that can be built, respectively;
- $D_{MIN}$  be the minimum distance between two turbines;
- $dist(i, j)$  be the symmetric distance between positions  $i$  and  $j$ .

In addition, let  $G_I = (V, E_I)$  denote the incompatibility graph with  $E_I = \{[i, j] : i, j \in V, dist(i, j) < D_{MIN}, j > i\}$ .

In our model, we define binary variables  $x_i$  for each  $i \in V$  to indicate whether a turbine is built in position  $i$  ( $x_i = 1$ ) or not ( $x_i = 0$ ). The quadratic objective function (to be maximized) reads  $\sum_{i \in V} P_i x_i - \sum_{i \in V} (\sum_{j \in V} I_{ij} x_j) x_i$  and can be restated as

$$\sum_{i \in V} (P_i x_i - w_i) \tag{1}$$

where

$$w_i := \left( \sum_{j \in V} I_{ij} x_j \right) x_i = \begin{cases} \sum_{j \in V} I_{ij} x_j & \text{if } x_i = 1; \\ 0 & \text{if } x_i = 0 \end{cases}$$

denotes the total interference caused by position  $i$ . Our MILP model then reads

$$\max \quad z = \sum_{i \in V} (P_i x_i - w_i) \quad (2)$$

$$\text{s.t.} \quad N_{MIN} \leq \sum_{i \in V} x_i \leq N_{MAX} \quad (3)$$

$$x_i + x_j \leq 1 \quad \forall \{i, j\} \in E_I \quad (4)$$

$$\sum_{j \in V} I_{ij} x_j \leq w_i + M_i (1 - x_i) \quad \forall i \in V \quad (5)$$

$$x_i \in \{0, 1\} \quad \forall i \in V \quad (6)$$

$$w_i \geq 0 \quad \forall i \in V \quad (7)$$

where the big-M terms  $M_i = \sum_{\substack{j \in V \\ \{i, j\} \in E_I}} I_{ij}$  are used to deactivate constraint (5) in case  $x_i = 0$ . Note that constraint (3) can be used to impose the construction of a fixed number of turbines by setting  $N_{MIN} = N_{MAX}$ . Note that the power production  $P_i$  and the interference value  $I_{ij}$  vary with the wind. Using statistical data, one can in fact collect a large number, say  $K$ , of wind scenarios  $k$ , each associated with its own  $P_i^k, I_{ij}^k$  and with a probability  $\pi_k$ . As shown in [5], one can take wind

scenarios into account in our model by simply defining  $P_i := \sum_{k=1}^K \pi_k P_i^k$  ( $i \in V$ ) and

$$I_{ij} := \sum_{k=1}^K \pi_k I_{ij}^k \quad (i, j \in V).$$

To solve large-scale instances (with 20000+ possible positions) some ad-hoc heuristics and a MILP-based proximity search [4] heuristic has been used on top of this basic model. We refer the interested reader to [5] for details.

### 3 Data Generation

We used the model presented in Sect. 2 to determine the optimized power production of a large number of realistic instances. These instances have been created by considering rectangular areas of different sizes, different turbine types, and different wind statistics from different real-world sites. In particular, we generated different sites by generating sets of possible points on a regular grid (10m point-to-point distance) inside rectangles of different dimensions (all possible combinations of edge sizes 6000, 7000, 8000, 9000, 10000, 11000, 12000, 13000 and 14000m). We computed power production and interference based on the data from the following real-world turbines:

- Adwen 8 MW, with a rotor diameter of 180 m;
- Vestas 8.4 MW, with a rotor diameter of 164 m;
- Siemens 7 MW, with a rotor diameter of 154 m;
- Vestas 8 MW, with a rotor diameter of 164 m;
- Siemens 3.2 MW, with a rotor diameter of 113 m;
- Siemens 2.3 MW, with a rotor diameter of 101 m.

Note that different rated powers and different rotor diameters affect the power production  $P_i$  and the interference  $I_{ij}$  of each turbine and therefore the total power production of the park. Finally, we considered different real-world wind statistics for the wind scenarios, namely from the real offshore wind parks named Borssele 1 and 2, Borssele 3 and 4, Danish Krigers Flak and Ormonde. These parks are in-operation or under-construction parks located in the Netherlands, Denmark, and the United Kingdom. By considering all the possible combinations of sites, turbine types and winds, we obtained 2268 instances. We imposed that a fixed number of 50 turbines need to be located in the site, at a minimum distance of 5 rotor diameters. For each instance we computed:

- (1) the so-called *gross production*, i.e., the power production of the optimized solution neglecting the interference factor (this is an upper bound of the optimized power production of the site);
- (2) the optimized layout and its power production.

Note that (1) requires very short computational time and can be calculated in a pre-processing step. Optimization for the difficult case (2) was instead obtained through the MILP-based heuristic of [5], with a time limit of 1 h on a standard PC using IBM ILOG CPLEX 12.6.

The output of the optimization has been used to train the ML models presented in Sect. 4, where the results of case (1) are considered as input features, while (2) is the figure that we aim at estimating.

## 4 Machine Learning

Feature definition is a key point in the development of ML models. In particular, we need to give to the ML model valuable information on the turbine type used, the wind and the site. In order to assess which is the most useful information to consider, we used our knowledge of the problem and different visualizations of the data. We concluded our analysis by selecting the following features:

- the rated power of the turbine: this is the maximum power (MW) that the turbine can produce (at high wind speeds); this feature describes the turbine model, and impacts both park production and interference;
- the rotor diameter of the turbine: this describes the dimension of the turbine and impacts the interference and the minimum distance between turbines;
- the gross power production: this captures the wind in the site;
- the area (in  $m^2$ ) of the site;
- the ratio between the two edges of the rectangle: this captures the shape of the site.

Note that, in order for our ML models to work, we need to encode our features in a way that is easy for the model to interpret. This is why, for example, we preferred to use the ratio between the edges instead of their individual length. In the same way, we did not explicitly pass the wind statistics of the site to the ML model, but we used instead the gross production (that gives richer information, as it relates the wind with the turbine type used).

Finally, instead of directly estimating the optimized production of a site, we estimate its normalized difference from the gross production, defined as

$$\text{reduction} = \frac{\text{gross production} - \text{optimized production}}{\text{gross production}} \quad (8)$$

This is a value between 0 and 1 that can easily be compared between instances with production of different scales.

We defined two different ML models to estimate the reduction in power production due to the interference, namely Linear Regression and Neural Networks (NNs). In addition, we also defined a simple baseline model (denoted in the following as *Mean Value*), that regardless of its input always predicts the mean reduction of the training set. This last model mimics what is normally done by humans, and is used for comparison.

We provided the previously described features on input to the ML models, organizing them in a vector  $\mathbf{x}$ . The estimated reduction in power production  $\hat{y}$  was then modelled through a function  $f$  that depends on some unknown parameters  $\mathbf{w}$  to be learned during the training phase, i.e.  $\hat{y} = f(\mathbf{w}, \mathbf{x})$ . We used the *Root Mean Squared Error (RMSE)* [3] to measure the quality of our ML models. RMSE gives a good description of the deviation between the predictions of the model and the true values, and is therefore widely used in the ML community to evaluate regression models. Given a training data set containing input-output pairs  $\{(\mathbf{x}_n, y_n), n = 1, \dots, N_{train}\}$ , the RMSE formula reads:

$$E(\mathbf{w}) = \sqrt{\frac{1}{N_{train}} \sum_{n=1}^{N_{train}} (y_n - \hat{y}_n)^2} = \sqrt{\frac{1}{N_{train}} \sum_{n=1}^{N_{train}} (y_n - f(\mathbf{w}, \mathbf{x}_n))^2}$$

where  $E(\mathbf{w})$  is minimized when our estimate  $\hat{y}_n$  is as close as possible to the “real value”  $y_n$ . The optimal parameters  $\mathbf{w}^*$  for our ML models are therefore found as  $\mathbf{w}^* = \arg \min_{\mathbf{w}} E(\mathbf{w})$ .

## 5 Preliminary Results

We defined the training set by randomly choosing 60% of the 2268 generated instances. The remaining 40% of the instances is only used for testing purposes (test set), and will give us a measure of how much our models *generalize* to previously-

**Table 1** RMSE of the test set for the different models

Model	RMSE
Mean value	0.0235
Linear regression	0.0101
Neural network	0.0059

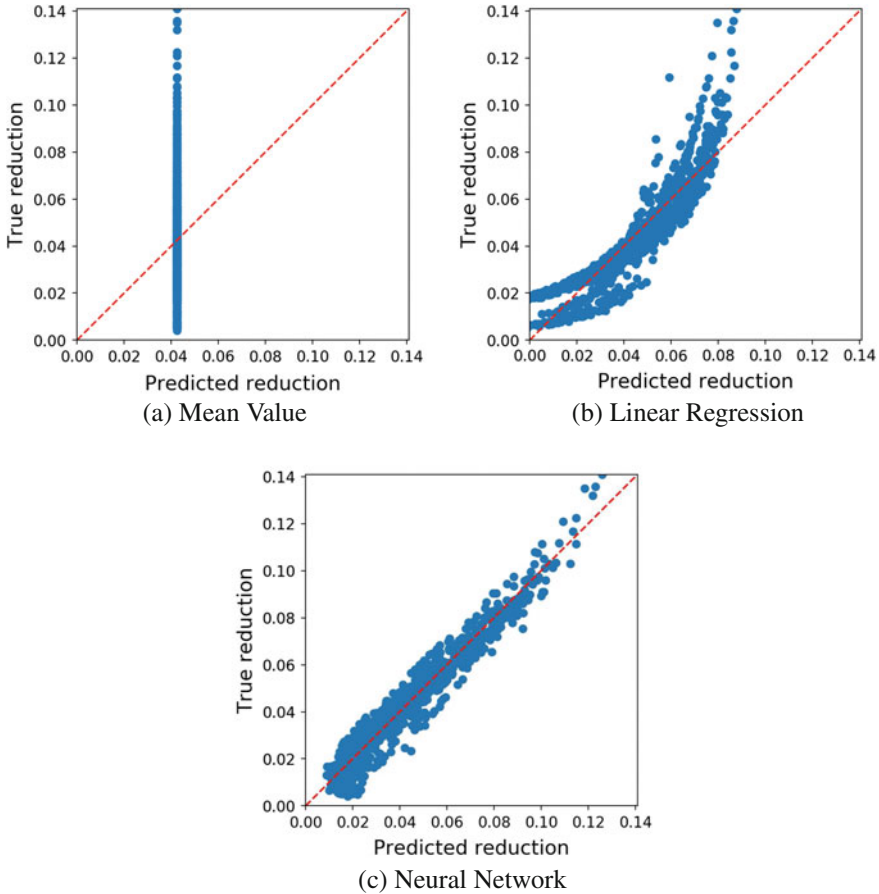
unseen data. As these models are very sensitive to different scaling of the input features, we standardize all the features to have mean 0 and standard deviation 1 over the training set.

For NNs we choose the different parameters (number of layers, number of hidden units, learning rate, amount of weight regularization and activation function) using the *scikit-learn* [7] function GridSearchCV, that exhaustively considers all parameter combinations on a grid (5-fold cross-validated on the training set). According to our tests, the best architecture is a single-layer neural network with 20 hidden units and hyperbolic tangent non-linearity. All the models were implemented using Python's machine learning library *scikit-learn* [7].

In Table 1 we compare the performance of the models in terms of RMSE on the test set. We see that both the baseline Mean Value estimator and Linear Regression are outperformed by the NN, suggesting that modelling non-linearities is fundamental for the task in hand. In Fig. 1 we visualize the test set predictions: on the  $y$ -axis we report the true reduction and on the  $x$ -axis its estimate from the model. Each point in the graph represents a test instance. If the predictions were perfect, all points should lay on the  $y = x$  line (in red in the plot). The Mean Value strategy predicts always the same value (so it appears as a vertical line in the plot) and fails in capturing the problem complexity. The Linear Regression strategy tends to underestimate the reduction (for low or high reduction values) or to overestimate it (for middle reduction values). NNs, instead, are able to estimate the reduction well (in the figure, all points are close to the  $y = x$  line). The comparison between NNs and Mean Value, in particular, shows the importance of using ML instead of a manual operator to analyse the data.

Note that all the training data comes from optimized solutions: without having a MO optimizer, the company would probably estimate the value of a site by considering suboptimal layouts, e.g., a layout with turbines on a regular grid. If we compare the production of a layout of 50 turbines on a regular grid (with 5-rotor-diameter distance between adjacent turbines) with the production of an optimized layout on the same site, the difference in our instances can be as high as 13%. This shows the importance of using MO models in the training phase.





**Fig. 1** Comparison between the predicted reductions  $\hat{y}$  (x axis) and the true reductions  $y$  (y axis) of the test set. The optimal predictions are shown with the dashed red line

## 6 Conclusions and Future Work

The present preliminary work showed the relevance of using MO and ML techniques together. We have shown that ML techniques (NNs in particular), trained on a large number of optimized solutions, could well predict the optimal value of new instances of the same problem. In this work, we have focused on the wind park layout problem.

A possible extension of the model could be to allow for different numbers of turbines or different shapes of the park site (not only rectangles). Finally, other ML models could be developed and compared on a larger dataset. More ambitiously, future work could investigate the application of our approach to different OR problems. One could, indeed, address the problem of estimating the optimal *value* of an optimization problem by using ML algorithms trained on optimized solutions com-

puted by time-consuming MO solvers. This estimate can be of interest by itself (as in the wind farm application studied in this paper), but can also be very useful, e.g., for heuristic node pruning in a branch-and-bound solution scheme.

## References

1. Barthelmie, R.J., Hansen, K., Frandsen, S.T., Rathmann, O., Schepers, J.G., Schlez, W., Phillips, J., Rados, K., Zervos, A., Politis, E.S., Chaviaropoulos, P.K.: Modelling and measuring flow and wind turbine wakes in large wind farms offshore. *Wind Ener.* **12**, 431–444 (2009)
2. Bertsimas, D., Kallus, N.: From Predictive to Prescriptive Analytics. [arXiv:1402.5481](https://arxiv.org/abs/1402.5481) (2014)
3. Bishop, C.M.: *Pattern Recognition and Machine Learning*. Information Science and Statistics, Springer, New York (2006)
4. Fischetti, M., Monaci, M.: Proximity search for 0–1 mixed-integer convex programming. *J. Heurist.* **20**(6), 709–731 (2014)
5. Fischetti, M., Monaci, M.: Proximity search heuristics for wind farm optimal layout. *J. Heurist.* **22**, 459–474 (2016)
6. Jensen, N.: A note on wind generator interaction. Technical Report Riso-M-2411(EN), Riso National Laboratory, Roskilde, Denmark (1983)
7. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: machine learning in python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)

# Mathematical Programming Bounds for Kissing Numbers

Leo Liberti

**Abstract** We give a short review of existing mathematical programming based bounds for kissing numbers. The *kissing number* in  $K$  dimensions is the maximum number of unit balls arranged around a central unit ball in such a way that the intersection of the interiors of any pair of balls in the configuration is empty. It is a cornerstone of the theory of spherical codes, a good way to find  $n$  equally spaced points on the surface of a hypersphere, and the object of a diatribe between Isaac Newton and David Gregory.

## 1 A Brit and a Scot Went Down the Pub...

“Kissing” is billiard jargon. British players would say two adjacent billiard balls on the table “kiss”. The term found its way into mathematics thanks to Isaac Newton (whom everyone knows) and David Gregory (a professor of Mathematics at Edinburgh—without having ever obtained a degree—and then Savilian Professor of Astronomy at Oxford thanks to Newton’s influence). In the 1690s, scared of the social unrest in Scotland, Gregory left and visited Newton in Cambridge. According to rumours and well-established British protocol, the brit and the scot went down the pub for a few pints of ale and a game of pool. There, among kissing balls and fumes of alcohol, they got into a brawl about the number of balls that could kiss a central ball on the billiard table. Still sober enough, they counted them, and came to agree on the number six. As the number of pints increased, the two started blabbering about gravity-defying floating balls passionately kissing in three dimensions, and disagreed: Newton, embracing the voice of Kepler, said no more than twelve balls could be arranged around a central one. Gregory, who had to gain his master’s approval by attempting to be brilliant and surprising, said that perhaps thirteen could fit? George Szpiro [20] recounts a different story

---

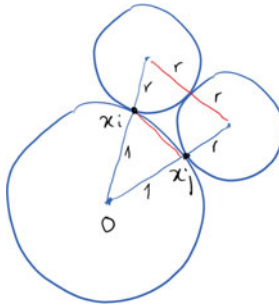
L. Liberti (✉)  
CNRS LIX Ecole Polytechnique, 91128 Palaiseau, France  
e-mail: liberti@lix.polytechnique.fr

in [plus.maths.org/content/newton-and-kissing-problem](http://plus.maths.org/content/newton-and-kissing-problem) (some nonsense about astronomy and planets), but since neither he nor I were present at the quabble, his word on the matter is just as good as mine.

## 1.1 Applications

Quite aside from the satisfaction I get out of spreading academic gossip on Isaac Newton, it turns out that arranging balls in a kissing configuration has applications.

If we only consider the points of contact of  $n$  surrounding balls of radius  $r$  with the central unit ball, we obtain a set of unit vectors  $x_1, \dots, x_n$  in  $\mathbb{R}^K$  that have pairwise distances at least  $\frac{2r}{1+r}$ . This can be seen in the figure on the left (a two-dimensional section of part of some  $K$ -dimensional balls configuration), together with the proportion  $2r : (1+r) = d_{ij} : 1$ , where  $d_{ij} = \|x_i - x_j\|_2$ . This is useful if you ever want to send one of the vectors  $x_i$  (for  $i \leq n$ ) over a noisy communication channel. You might receive a vector  $y$  different from all  $x_i$ 's, but assuming the channel is not too noisy, you can simply assume that  $y$  is a corruption of the closest  $x_i$ . This type of error correcting code is called a *spherical code*, and denoted by  $A(n, K, r)$ . There is interest in maximizing  $r$ , since this corresponds to larger balls and consequently more errors being corrected by the code. Kissing number configurations correspond to spherical codes  $A(n, K, 1)$  where  $n$  is maximum for a given  $K$ . Finding a spherical code given  $n, K, r$  is the SPHERICAL CODE PROBLEM (SCP).



The other (less cited) application is finding  $n$  equally spaced points on the  $K$ -dimensional sphere  $S^{K-1}$ . On websites such as <http://stackoverflow.com> or <http://math.stackexchange.com>, it seems people expect this to be an easy problem (possibly because the circle is a simple case: place  $x_i$  on the circle at an angle  $2i\pi/n$ ). Some 3D solutions advise scattering equally spaced points on a spiral going from one pole to the opposite, warning readers that it does not guarantee equal spacing. The problem can be formulated for any  $K$  by means of spherical codes where the smallest  $r$  is maximum. Finding  $n$  equally spaced unit vectors on  $S^{K-1}$  is the EQUALLY SPACED SPHERICAL POINTS PROBLEM (ESSPP).

## 1.2 Contents

Most of the results in this paper are known. I propose a new SDP-based heuristic for finding lower bounds, and I give a practitioner's view of Delsarte's Linear Programming (LP) upper bound [7], which is useful for conducting experiments in view of trying to improve current bounds ([2, 14] also discuss practical issues of computing these bounds). All the experiments reported in this paper are preliminary. Lastly, I enjoyed writing this paper more informally than is usual—I hope readers won't object!

## 2 The Kissing Number Problem

Finding kissing numbers and kissing configurations is known as the KISSING NUMBER PROBLEM (KNP). Formally, this consists in finding the maximum number  $n$  of vectors  $x_1, \dots, x_n \in \mathbb{R}^K$  such that  $\|x_i\|_2^2 = 1$  for all  $i \leq n$  and  $x_i \cdot x_j \leq \frac{1}{2}$  for all  $i < j \leq n$ . If  $n$  is the kissing number in  $K$  dimensions, we write  $\text{kn}(K) = n$ . We know that  $\text{kn}(2) = 6$ ,  $\text{kn}(3) = 12$  (so Newton was right),  $\text{kn}(4) = 24$ , and we do not know  $\text{kn}(5)$  but it is at least 40. We also know  $\text{kn}(8) = 240$  and  $\text{kn}(24) = 196560$  [4, p. 510], and have bounds in many other dimensions (see [https://en.wikipedia.org/wiki/Kissing\\_number\\_problem](https://en.wikipedia.org/wiki/Kissing_number_problem)).

### 2.1 Computational Complexity

The computational complexity of solving KNP, as an optimization or even a decision problem, is a prominent and embarrassing question mark. Since the input is a pair of integers  $n, K$ , mapping an NP-complete problem (e.g. SAT) to a KNP, such that an instance of one problem is yes if and only if the corresponding instance of the other problem is also yes, really seems quite hard.

On the other hand, the KNP is certainly very hard to solve empirically, and no-one working on this problem ever suggested that there might be a polynomial-time algorithm for solving it—even on a real RAM computational model.

Reductions between decision versions of KNP, SCP and ESSPP are as follows: the KNP is included in the SCP by definition (so it trivially reduces to the SCP), and, as pointed out in Sect. 3.2, the KNP can be decided by the decision version of the ESSPP (so, again, it reduces to the ESSPP). The decision versions of the SCP and ESSPP are really the same (though the optimization versions differ). The only reduction I cannot immediately prove is from SCP/ESSCP to the KNP, since only the latter fixes the angular separation at  $\pi/3$ .

I am not aware of *any method* for proving NP-hardness of problems whose input consists of a constant number of integers. For example, the complexity status of the

well-known PACKING EQUAL CIRCLES IN A SQUARE (PECS) problem is as yet undetermined [6]. This is certainly an interesting open problem in computational complexity.

A referee pointed out an interesting link with another problem having undetermined complexity status: the PALLET LOADING PROBLEM (PLP), which asks whether  $n$  identical  $a \times b$  rectangles can be packed in a given  $X \times Y$  rectangle [12, §2.C]. Thus, PLP instances, like KNP ones, are described by a constant number of integers. Again, establishing reductions between KNP, SCP, ESSPP, PECS and PLP is an open question.

### 3 Lower Bounds

Since the KNP is a maximization problem, the cardinality of any kissing configuration of balls yields a lower bound. Since any heuristic method might be able to find a good configuration, lower bounds for the KNP are considered “easy” to obtain.

#### 3.1 The Formulation of Maculan, Michelon and Smith

We start with the MMS95 formulation proposed in [11], a Mixed-Integer Nonlinear Program (MINLP) which correctly formulates the KNP:

$$\left. \begin{aligned} \max_{x \in [-1,1]^{\tau K}, \alpha \in \{0,1\}^{\tau}} \quad & \sum_{i=1}^{\tau} \alpha_i \\ \forall i \leq \tau \quad & \|x_i\|_2^2 = \alpha_i \\ \forall i < j \leq \tau \quad & \|x_i - x_j\|_2^2 \geq \alpha_i \alpha_j, \end{aligned} \right\} \tag{1}$$

where  $\tau$  is some (estimated) upper bound to the kissing number. The  $\alpha_i$  (binary) variables choose whether vector  $x_i$  is part of the configuration or not. Note that the angular separation constraint  $x_i \cdot x_j \leq \frac{1}{2}$  is replaced by a Euclidean distance separation  $\|x_i - x_j\|_2^2 \geq 1$ , which is equivalent since  $1 \leq \|x_i - x_j\|_2^2 = \|x_i\|_2^2 + \|x_j\|_2^2 - 2x_i \cdot x_j = 2 - 2x_i \cdot x_j$  (whence  $x_i \cdot x_j \leq 1/2$ ) as the vectors all have unit norm. Since the problem is formulated exactly, an exact MINLP solver would provide a feasible and optimal solution if it exists, given some guessed upper bound.

Unfortunately, the state of the art in MINLP solver technology cannot even provide an answer in  $K = 2$  if  $\tau = 7$  (one more than  $\text{kn}(2) = 6$ ) in “reasonable time” of a “reasonable laptop” using Eq. (1).

### 3.2 A Feasibility Formulation

Given  $K$  and  $n$ , the formulation below finds the configuration of  $n$  unit vectors where the minimal separation between closest vectors is maximum [10]:

$$\left. \begin{aligned} \max_{x \in [-1,1]^{nK}, \alpha \geq 0} \quad & \alpha \\ \forall i \leq n \quad & \|x_i\|_2^2 = 1 \\ \forall i < j \leq n \quad & \|x_i - x_j\|_2^2 \geq \alpha. \end{aligned} \right\} \tag{2}$$

Equation (2) has a single scalar  $\alpha$  variable, which is continuous and represents the minimum distance between pairs of vectors. Solving this nonconvex Nonlinear Program (NLP) to global optimality and obtaining an optimal  $\alpha \geq 1$  yields a proof that  $\text{kn}(K) \geq n$ ; if  $\alpha < 1$  then  $\text{kn}(K) < n$ . The issue with this strategy for proving kissing numbers is the same as for Eq. (1): current solvers just cannot solve these instances to global optimality for interesting values of  $n, K$ .

On the other hand, Eq. (2) can be used heuristically to find KNP configurations given  $n, K$ . Moreover, these feasible solutions will in general spread points over the  $K$ -sphere quite evenly, each point being at roughly the same distance from its closest points. They will therefore provide a practical solution to the ESSPP.

### 3.3 Semidefinite Programming Relaxations

Semidefinite Programming (SDP) relaxations of Eqs. (1) and (2) have not been looked at yet, as far as I know. I performed a few preliminary tests on both, and while the SDP from Eq. (1) seems extremely slack (trivially yielding the upper bound  $\tau$  for whatever given  $\tau$ , probably due to relaxed integrality), I found the SDP relaxation of Eq. (2) more interesting:

$$\left. \begin{aligned} \max_{X \in [-1,1]^{n^2}, \alpha \geq 0} \quad & \alpha \\ \forall i \leq n \quad & X_{ii} = 1 \\ \forall i < j \leq n \quad & X_{ii} + X_{jj} - 2X_{ij} \geq \alpha \\ & X \geq 0. \end{aligned} \right\} \tag{3}$$

Based on past experience with other problems involving Euclidean distance constraints, I tried the following heuristic strategy:

1. solve Eq. (3) and obtain an optimal solution  $(\bar{X}, \bar{\alpha})$ ;
2. perform Principal Component Analysis (PCA) using the  $K$  largest eigenvalues of  $\bar{X}$  to obtain an  $n \times K$  matrix  $\bar{x}$  such that the  $i$ -th row  $\bar{x}_i$  is a vector in  $\mathbb{R}^K$ ;
3. use  $\bar{x}$  as a starting point for a local NLP solver deployed on Eq. (2), obtain a “good” feasible solution  $(x^*, \alpha^*)$ .

Using Mosek [15], a Python implementation of PCA, and IPOPT [5], I was able to derive the following KNP configurations on my “reasonable laptop” based on a 3.1 GHz Intel Core i7 with 16 GB RAM.

$(n, K)$	(6, 2)	(12, 3)	(24, 4)	(40, 5)	(72, 6)
$\varepsilon$	0	0	0.04	0.05	0.07
CPU (s)	0.02	0.02	0.32	1.57	12.26

The first row measures the (additive) solution error with respect to a valid KNP configuration:  $\alpha^* = 1 - \varepsilon$  for all  $\alpha < 1$  (whereas  $\alpha^* > 1$  whenever  $\varepsilon = 0$ ). Unfortunately, the next interesting case,  $n = 12$  and  $K = 7$ , made Mosek crash for lack of RAM (the computational bottleneck on solving SDPs is well known).

## 4 Upper Bounds

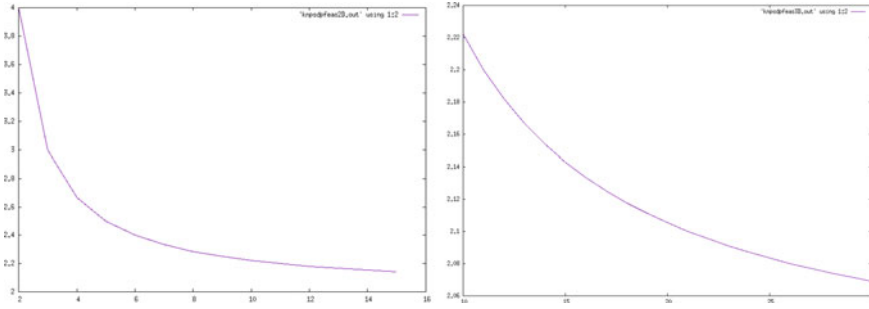
In general, upper bounds to the KNP are considered hard to obtain. Each new upper bound either requires a completely new theoretical point of view, or a substantial body of theoretical work and some computation.

### 4.1 Direct LP and SDP Bounds

A couple of easy upper bounding techniques, however, are readily available in the optimization literature: LP and SDP relaxation. Expanding the Euclidean distances in Eqs. (1) and (2) yields a MINLP and, respectively, a nonconvex NLP involving products of decision variables as the only type of nonlinearity. Using McCormick’s [13] for bilinear products and the secant relaxation for square terms one can obtain an LP. For having tested it in the past while I worked on [10], I know that this kind of LP bound is as slack as they come.

Based on some preliminary experiments using the Mosek SDP solver [15] on Eq. (3) for  $K \in \{2, 3\}$ , I observed a regular decrease in the optimal value  $\bar{\alpha}$  of the objective function of Eq. (3) as  $n$  increases for a given fixed  $K$ , as shown in the figures below.





The decreasing sequence of optimal  $\bar{\alpha}$  values appears to be converging to 2 for both  $K = 2$  and  $K = 3$ . I have not even started considering how to prove it (or disprove it), nor whether it would have any interesting consequence.

### 4.2 Delsarte’s LP Bound

This is an adaptation to the spherical context of Delsarte’s upper bound technique for binary codes [8], based on LP.

Broadly speaking, Delsarte’s idea is based on deciding the distance distribution of the code so as to maximize its cardinality. Let  $a_t$  be the fraction of the vectors in a KNP configuration code  $A(n, K, 1)$  that have scalar product equal  $t$ . Since these codes contain  $n$  vectors, there are at most  $n^2$  values of  $t$  (there could be fewer if many scalar products have the same value). In any case, summing over the  $a_t$  we obtain  $n^2/n = n$ , which is the cardinality of the code. Also, since these are fractions,  $a_t \geq 0$ . Moreover, there are exactly  $n$  scalar products having value 1, namely  $x_i \cdot x_i = \|x\|_2^2 = 1$  for all  $i \leq n$ , thus  $a_1 = n/n = 1$ . Hence, the LP

$$\max \left\{ \sum_t a_t \mid a_1 = 1 \wedge a \geq 0 \right\} \quad (\dagger)$$

is of interest to us, insofar as it maximizes the cardinality of the code it describes. The issue at this point is that this LP is unbounded—there is nothing that links the decision variables  $a_t$  to the “code structure”. Delsarte’s idea consists in finding a family  $\mathcal{F} = \{\phi_1, \phi_2, \dots\}$  of functions  $\phi : [-1, 1] \rightarrow \mathbb{R}$  such that

$$\forall \phi \in \mathcal{F} \quad \sum_t a_t \phi(t) \geq 0 \quad (\ddagger)$$

is a valid constraint for the LP  $(\dagger)$ .

Delsarte’s “main theorem” has been proved many times, and generalized in various ways [4, §2.2]. Its statement is also valid for the SPC and the ESSPP, as it considers an arbitrary separation angle  $\zeta$  with  $\cos \zeta = z$ .

**Theorem 1** *Let  $c_0 > 0$  and  $f : [-1, 1] \rightarrow \mathbb{R}$  such that:*

- (i)  $\sum_{i,j \leq n} f(x_i \cdot x_j) \geq 0$
- (ii)  $\forall t \in [-1, z] f(t) + c_0 \leq 0$
- (iii)  $f(1) + c_0 \leq 1$ .

Then  $n \leq \frac{1}{c_0}$ .

My favorite proof is the one-liner given in [18]. Let  $g(t) = f(t) + c_0$ , then:

$$n^2 c_0 \leq n^2 c_0 + \sum_{i,j \leq n} f(x_i \cdot x_j) = \sum_{i,j \leq n} g(x_i \cdot x_j) \leq \sum_{i \leq n} g(x_i \cdot x_i) = n g(1) \leq n,$$

whence  $n \leq 1/c_0$ . This suggests that the problem  $\max\{c \mid \text{(i)–(iii)}\}$  (\*) is relevant, as higher values for  $c_0$  correspond to tighter bounds. We look for a function  $f$  written as a linear combination of functions  $\phi_h \in \mathcal{F}$ , and introduce the coefficients  $c_h$  so that  $f(t) = \sum_h c_h \phi_h(t)$ .

We now come to the family  $\mathcal{F}$ : Delsarte’s LP bounding techniques require an orthogonal family of polynomials. In the KNP case, this family consist of *Gegenbauer polynomials*  $C_h^{(\lambda)}(t)$  [1, p. 776], that encode certain properties of  $S^{K-1}$ . For example, so far our description of Delsarte’s LP has failed to include any information about  $K$ , which is provided by this choice of  $\mathcal{F}$ , notably by setting  $\lambda = (K - 2)/2$  [19, §3]. Another crucial property is that if a function  $f$  is a conic combination of Gegenbauer polynomials, then  $(f(x_i \cdot x_j))_{ij} \geq 0$ , whence condition (i) of Theorem 1 holds; moreover, conversely, *any* function satisfying  $(f(x_i \cdot x_j))_{ij} \geq 0$  can be written as a conic combination of Gegenbauer polynomials [19]. We therefore adjoin the constraints (‡) quantified over  $\mathcal{G}^K = \{C_h^{(K-2)/2}(t) \mid h \in H\}$  (for some set  $H$ ) to the LP (†).

We explicitly write the LP (\*) by requiring that the  $f$  appearing in Theorem 1 should be a conic combination of elements of  $\mathcal{G}^K$ :

$$\max_{c_0 \geq 0} \{c_0 \mid \forall t (f(t) = \sum_{\phi_h \in \mathcal{G}^K} c_h \phi_h(t) \wedge c_0 + f(t) \leq 0) \wedge c_0 + f(1) \leq 1\}. \quad (4)$$

As observed in [17], (†) and (\*) are a pair of dual LPs. You have to “massage” (\*) somewhat before the duality relation with (4) becomes apparent, though (eliminate  $c_0$  and minimize  $\sum_h c_h$ , see [2, Eq. (5), p. 615]). By duality, we only focus on one LP, namely Eq. (4).

### 4.2.1 Getting Your Hands Dirty

As stated, Eq. (4) is infinite in both dimensions:  $\mathcal{F}$  is countably infinite (if we restrict  $\lambda$  to be integer) and  $t$  varies in the uncountably infinite set  $\bar{T} = [-1, 1/2] \cup \{1\}$ . Only a couple among the many works in the literature mention this as a difficulty (without

providing a discussion, however). The only paper I found that provides a satisfactory discussion of this issue is [2].

1. For an upper bound to  $\text{kn}(K)$ , start with polynomials in  $\mathcal{G}^K$  up to degree, say, 15 [17], and gradually work your way up the degrees to see if the bounds improve (be wary of floating point errors arising from evaluating polynomials large degrees—they may invalidate the bound, see [2, Lemma 1]). You can also use any polynomial from  $\mathcal{G}^\ell$  for  $\ell \geq K$  (though not for  $\ell < K$ ), since Gegenbauer polynomials having lower  $\lambda$  “dominate”—in the sense that they yield larger values of  $c_0$ —those for higher  $\lambda$ . Essentially, the minimum  $\ell$  for which you choose elements in  $\mathcal{G}^\ell$  determines the dimensionality of the KNP instance you are going to compute a bound for.
2. A safe way to deal with the fact that  $\bar{T}$  is infinite is to choose any finite  $T \subseteq \bar{T}$ , as removing constraints yields a relaxation (and hence a valid bound). A few experiments will convince you that this discretization has a very small impact on the value of the bound—moreover, since these are “small LPs”, you can still instantly obtain answers even when  $|T|$  is pretty large (e.g.  $O(10^5)$ ).
3. In order to obtain closed form expressions for Gegenbauer polynomials, I used Mathematica’s `GegenbauerC[h, λ, t]`. However, Eq. (4) expects Gegenbauer polynomials to be normalized so that  $C_h^\lambda(1) = 1$ , whereas Mathematica’s implementation normalizes them differently—you have to remember to evaluate `GegenbauerC[h, λ, t] / GegenbauerC[h, λ, 1]`.

I obtained the following (standard) LP upper bounds using Gegenbauer families  $C_h^{0.5}$  and  $C_h^1$  respectively, for  $h \leq 10$ :  $\text{kn}(3) \leq \lfloor 13.1583 \rfloor$ ,  $\text{kn}(4) \leq \lfloor 25.5581 \rfloor$  and  $\text{kn}(5) \leq \lfloor 46.3365 \rfloor$ .

### 4.3 Extensions

I would like to emphasize three among the extensions to Delsarte’s LP bound:

- (a) Oleg Musin’s extension [16], which brought us the proof that  $\text{kn}(4) = 24$ ;
- (b) Pfender’s extension [18], yielding new upper bounds for  $K \in \{10, 16, 17, 25, 26\}$ ;
- (c) Bachoc and Vallentin’s extension [3], which brought us  $\text{kn}(5) \leq 45$  (further improved to  $\text{kn}(5) \leq \lfloor 44.99899685 \rfloor = 44$  in [14], obtained using the GNU Multiple Precision (GMP) arithmetic library: if you trust the GMP library, you should also trust this bound.

The first two extensions are based on enriching the function family  $\mathcal{F}$ : in case (a) by including some polynomials  $f$  such that condition (ii) of Theorem 1 is violated for a fixed  $x_i$  and finitely many  $x_j$  (the exceptions that ensue are dealt with combinatorially); and in case (b) by adding a new family of functions to  $\mathcal{F}$  that are not convex combinations of Gegenbauer polynomials but satisfy the conditions of Theorem 1. The case (c) is even more interesting as it considers distance distributions on

*triplets* of vectors (rather than pairs) and derives a semidefinite programming (SDP) formulation instead of an LP.

See [4] for further extensions.

## References

1. Abramowitz, M., Stegun, I. (eds.) Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, vol. 55, 10th edition. In: National Bureau of Standards, Applied Mathematics Series. US Government Printing Office, Washington (1972)
2. Antreichner, K.: The thirteen spheres: a new proof. *Discrete Comput. Geom.* **31**, 613–625 (2004)
3. Bachoc, C., Vallentin, F.: New upper bounds for kissing numbers from semidefinite programming. *J. Am. Math. Soc.* **21**, 909–924 (2008)
4. Boyvalenkov, P., Dodunekov, S., Musin, O.: A survey on the kissing numbers. *Serdica Math. J.* **38**, 507–522 (2012)
5. COIN-OR. Introduction to IPOPT: a tutorial for downloading, installing, and using IPOPT (2006)
6. Costa, A., Hansen, P., Liberti, L.: On the impact of symmetry-breaking constraints on spatial branch-and-bound for circle packing in a square. *Discrete Appl. Math.* **161**, 96–106 (2013)
7. Delsarte, P., Goethals, J.M., Seidel, J.J.: Spherical codes and designs. *Geom. Dedicata.* **6**, 363–388 (1977)
8. Delsarte, P.: Bounds for unrestricted codes by linear programming. *Philips Res. Rep.* **27**, 272–289 (1972)
9. Dias, G., Liberti, L.: Diagonally dominant programming in distance geometry. In: Cerulli, R., Fujishige, S., Mahjoub, R. (eds.) International Symposium in Combinatorial Optimization. LNCS, vol. 9849, pp. 225–236. Springer, New York (2016)
10. Kucherenko, S., Belotti, P., Liberti, L., Maculan, N.: New formulations for the kissing number problem. *Discrete Appl. Math.* **155**(14), 1837–1841 (2007)
11. Maculan, N., Michelon, P., Smith, J.M.: Bounds on the kissing numbers in  $\mathbb{R}^n$ : mathematical programming formulations. Technical report, University of Massachusetts, Amherst, USA (1996)
12. Martins, G.: Packing in two and three dimensions. PhD thesis, Naval Postgraduate School (2003)
13. McCormick, G.P.: Computability of global solutions to factorable nonconvex programs: Part I—convex underestimating problems. *Math. Program.* **10**, 146–175 (1976)
14. Mittelman, H., Vallentin, F.: High-accuracy semidefinite programming bounds for kissing numbers. *Exp. Math.* **19**(2), 175–179 (2010)
15. Mosek ApS.: The mosek manual, Version 8. <http://www.mosek.com> (2016)
16. Musin, O.: The kissing number in four dimensions. *Ann. Math.* **168**, 1–32 (2008)
17. Odlyzko, A., Sloane, N.: New bounds on the number of unit spheres that can touch a unit sphere in  $n$  dimensions. *J. Comb. Theory A* **26**, 210–214 (1979)
18. Pfender, F.: Improved delarte bounds for spherical codes in small dimensions. *J. Comb. Theory A* **114**(6), 1133–1147 (2007)
19. Schoenberg, I.: Positive definite functions on spheres. *Duke Math. J.* **9**(1), 96–108 (1942)
20. Szpiro, G.: Newton and the kissing problem. Plus magazine (online), 23 January 2003

# Learning Greedy Strategies at Secondary Schools: An Active Approach

Violetta Lonati, Dario Malchiodi, Mattia Monga  
and Anna Morpurgo

**Abstract** We describe an extra-curricular learning unit for students of upper secondary schools, focused on the discovery of greedy strategies. The activity, based on the constructivistic methodology, starts by analyzing the procedure *naturally* arising when we aim at minimizing the total number of bills and coins used for giving change. This procedure is used as a prototype of greedy algorithms, whose strategies are formalized and subsequently applied to a more general scheduling problem with the support of an ad hoc developed software.

**Keywords** CS teaching · Active teaching · Greedy algorithms

## 1 Introduction

After several dark decades during which school pupils were often forced to identify informatics with mere dexterity with ICT tools, the education systems are now starting to reconsider the importance of the core aspects of the discipline. Many initiatives [4, 5, 13, 14, 16] have worked hard to change the general perception and it is now ongoing an explicit effort to bring into play key informatic concepts such as abstraction, logic, data representation and a general attitude to ‘*computational thinking*’ [6, 8], i.e. to think about problems and their solutions in a way suitable to automatic processing. In this context, algorithms are becoming a very important

---

V. Lonati · D. Malchiodi (✉) · M. Monga · A. Morpurgo  
Dipartimento di Informatica, Università degli Studi di Milano,  
via Comelico 39, 20135 Milan, Italy  
e-mail: dario.malchiodi@unimi.it

V. Lonati  
e-mail: violetta.lonati@unimi.it

M. Monga  
e-mail: mattia.monga@unimi.it

A. Morpurgo  
e-mail: anna.morpurgo@unimi.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_23

topic, and there is a need to design new activities to foster learning how to reason on non trivial algorithms. Optimization algorithms, with their compelling usefulness, are very good candidates. Thus, we started thinking on potential activities with them, and we decided to focus on greedy strategies, well suited even for pupils of non vocational secondary schools.

In fact, a greedy strategy is a very *natural* way to cope with optimization problems: its short-sighted structure is attractive even for untrained minds, often uncomfortable with more elaborate planning of computing steps. However, while intuitive to adopt, a greedy solution is not always optimal and choosing the right greedy criterion (and convincing oneself that the choice is indeed optimal) is much more difficult and “unnatural”. We decided, however, that this could be an important computational thinking learning objective. Thus, by following the chapter on greedy algorithms in the book by Jon Kleinberg and Éva Tardos [9], we developed a learning unit on greedy strategies for upper secondary schools based on constructivistic approach, and in particular using our methodology called *algomotricity* [1–3, 11, 12].

The paper is organized as follows: in Sect. 2 we sketch our methodological root, in Sect. 3 we describe the learning unit, and in Sect. 4 we draw some conclusions.

## 2 Constructivistic Learning Theory and Computer Science Education

Constructivist learning theory, which has its origins in Piaget and Vygotsky [15, 17], states that people construct their own understanding and knowledge of the world through experiencing things and reflecting on those experiences. The learners are the creators of their own knowledge, and the learning process relies to a large extent on what they already know and understand; when faced with something new they need to reconcile it with their current mental schemes and conceptions.

Starting from this assumption, the question of what stimuli and tools, methods and strategies are more effective is fundamental. If knowledge cannot be simply *transferred* but must be *constructed*, its acquisition should be an individually tailored process, where learners are in charge of the learning process and the role of teachers is to create suitable contests and materials that favor such process, accompany the learners’ discoveries, and promote a metacognitive reflection about what they are doing and how their understanding is developing.

Work in small groups has the power to foster cognitive development and thus to empower learning. The group is a place to belong to and as such it provides support and motivation; it promotes cooperation and the activation of latent cognitive potentials through the sharing of different competences and working/thinking styles; it supports co-construction of knowledge by socialisation and reciprocal negotiation of meanings. Through the socio-cognitive conflict that emerges in groups, learners have the opportunity and the need to explain, confute and defend their beliefs; new aspects and prospects can be seen; personal experiences and point of views can be downsized and put in perspective.

<p><b>Knowledge</b></p> <ul style="list-style-type: none"><li>• Definition of optimization problems.</li><li>• Outline of greedy strategies and their limits.</li><li>• Approaches to analyze the correctness of a greedy algorithm.</li></ul> <p><b>Skills</b></p> <ul style="list-style-type: none"><li>• Describing a greedy strategy/procedure.</li><li>• Executing a greedy algorithm.</li><li>• Establishing if a greedy algorithm finds an optimal solution on a given input of small size.</li><li>• Establishing if an instance provides a counterexample for optimality of a greedy algorithm.</li></ul> <p><b>Competences</b></p> <ul style="list-style-type: none"><li>• Searching for a counterexample to disprove a property, or general reasons/proofs to conclude that a property holds.</li></ul>
--

**Fig. 1** Learning goals of the unit on greedy algorithm

This approach is especially fruitful to develop competences in problem solving, which are usually difficult to acquire by means of explanations and examples only. On the contrary, the learning process can be activated when facing a problem for which one's own repertoire of known procedures is insufficient [7, 10]: with exploration and discovery activities in small working groups, learners can learn together what they need to know in order to solve the problem.

Having these premises in mind, since 2011 we are experimenting with active workshops sharing a common strategy, which we call *algotricity*. As the name suggests (a portmanteau combining *algorithm* and *motoric*), our approach exploits kinesthetic learning activities, having the aim of informally exposing students to a specific informatics topic, followed by an abstract learning phase devoted to let students build their mental models of the topic under investigation and a final computer-based phase to close the loop with their previous acquaintance with applications. Our activities start “unplugged”, but they always end with work in which students are confronted with specially conceived pieces of software in order to make clear the link (but also the intellectual hierarchy) with the computing technology.

The learning goals of the unit are summarized in Fig. 1.

### 3 The Learning Unit

The learning unit is organised in two main phases. In the first one students work on an algorithm for giving change using the minimum number of available coins and bills; in the second one they are faced with a scheduling problem that can be tackled

by using a simple software tool we developed ad hoc.<sup>1</sup> Most activities are carried out actively by students; only between the two phases and at the end of the unit some taught explanations are given.

### ***3.1 Giving the Change***

The class is faced with the (simple) task of giving change using coins and bills. Some examples are computed with the whole class, in order to make evident that everybody is able to accomplish such a task easily, with no need of deep reasoning. The question then is asked whether the number of bills/coins used to give the change is as small as possible: usually students have no doubts that this is the case.

After this introduction, they are asked to work in pairs to put in written words the procedure to compose a given amount by using the smallest number of bills/coins. To avoid the use of technical jargon or constructs (what occurs for instance with students who already have some programming background) and in order to make the required level of detail clear, students are invited to “write the procedure so that it can be executed by a 10 years old child who is familiar with basic arithmetic operations”. A set of play money can be used to help formalize the procedure and to test it through a step-by-step execution.

Most groups usually propose a procedure that manages to accumulate the change through subsequent selections of one coin/bill having the highest value yet not exceeding the residual change; some use a more succinct approach exploiting the quotient and remainder of the division between the residual change and a coin/bill nominal value. Some explicit an initial step that sorts coins and bills according to their value, some leave such natural/obvious sorting implicit.

Working pairs are then merged into groups of 4–6 people so that pairs whose procedures share a common approach are put together. Each group is required to socialize the procedures of the pairs merged in the group, and agree upon a (possibly new) common procedure.

When all are ready, each group, in turn, reads its algorithm aloud, while the remaining ones test it. Remarks are welcomed, both on the features of the presented procedures and on their commonalities or differences. Starting from these remarks, the conductor draws and highlights the commonalities and proposes the unifying schema in Fig. 2, while ascertaining that students recognize their own procedures.

---

<sup>1</sup>The scheduling software is available at <http://aladdin.unimi.it/sw/scheduling/scheduling.html> (in Italian).



- |  |
|--|
| <ol style="list-style-type: none"> <li>1. Sort coins and bills in decreasing order of their nominal value;</li> <li>2. For each bill/coin, namely <math>X</math>, taken in that order:             <ul style="list-style-type: none"> <li>if the value of <math>X</math> is not higher than the residual change</li> <li>then use it,</li> <li>else reject it (and never consider again bills/coins with this value).</li> </ul> </li> </ol> |
|--|

Fig. 2 General schema for a greedy strategy

<p><b>Optimization problem</b></p> <p>Instance: a set <math>A</math> of objects</p> <p>Admissible solutions: any subset <math>S \subseteq A</math> satisfying the validity constraint</p> <p>Optimal solution(s): an admissible solution with minimal cost or maximal value</p> <p><b>Greedy procedure</b></p> <ol style="list-style-type: none"> <li>1. Sort objects according to some criterion;</li> <li>2. For each object, namely <math>X</math>, taken in that order:             <ul style="list-style-type: none"> <li>if <math>X</math> satisfies the validity constraint:</li> <li>then use it,</li> <li>else reject it (and never consider it again).</li> </ul> </li> </ol>
---

Fig. 3 General form of optimization problems and general outline for greedy strategies

### 3.2 Outline of Optimization Problems and Greedy Strategies

At this point, the conductor can generalize such approach to a more abstract procedure for a generic optimization problem which builds the solution by considering a set of objects in a given order, and for each of them decides whether or not it has to be enlisted in the solution according to a validity constraint (see Fig. 3). It should be emphasized that this constitutes an example of *greedy procedure*, in that each object is considered only once for (possibly multiple) addition to the solution, without any possibility to remove objects previously added to the solution.

### 3.3 A Scheduling Problem

The second part of the unit consists in asking the students to apply such approach to a scheduling problem, namely that of maximizing the number of movies to be seen in a film festival whose program contains several, partially overlapping movies. First, students are guided to find the analogies between this problem, the one concerning money change and the abstract description of a greedy procedure. Table 1 highlights such analogies: for instance, students tend to easily spot that movies, as well as coins/bills, play the role of objects. Analogously, the validity constraints are that of checking whether: (i) a movie does not overlap with the ones already in the solution, and (ii) the considered coin/bill has a nominal value smaller than the residual change. Once the analogies have clearly emerged, the discussion can be focused

**Table 1** Comparison between the scheduling and money change problems

	Change problem	Scheduling problem
Objects	Coins/bills	Movies
Sorting order	Nominal value, decreasing	Several alternatives
Validity constraint	Value not greater than the residual change	No overlapping between one movie and the ones already added

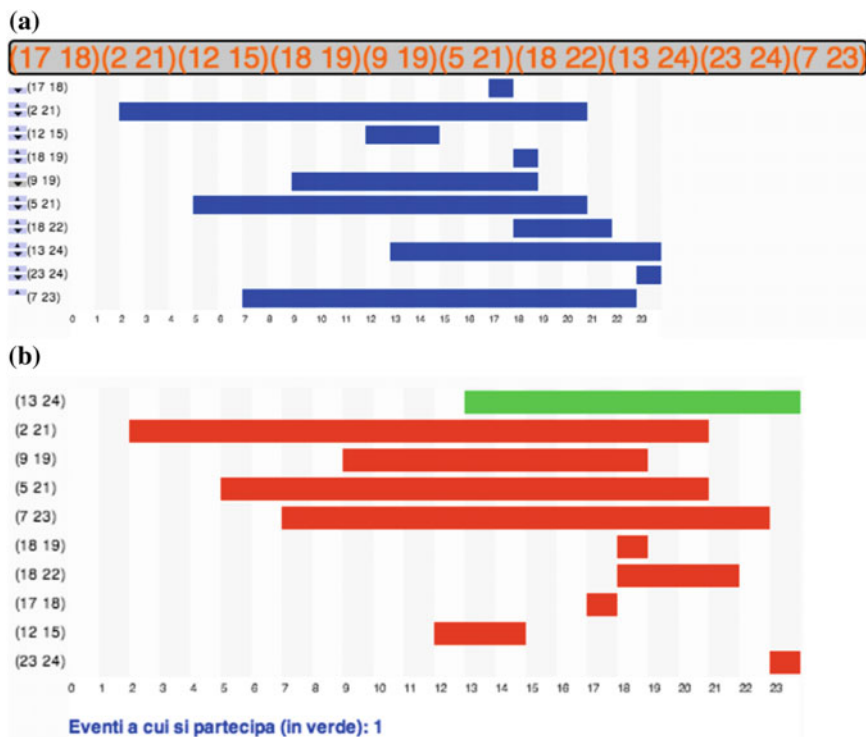
on the feasibility of a greedy approach to find an optimal solution for the scheduling problem. The main effort here is to figure out how to sort the movies, what instead was obvious in the case of coins and bills. Brainstorming should be suggested in order to collect several possible sorting criteria; the following are usually spotted: starting or ending time, number of intersections with the other movies, or movies' length (either ascending or descending).

Once all alternatives are clear, students are asked to work in pairs to verify which criteria ensure the greedy procedure to maximize the number of seen movies. In particular, students are asked to find counterexamples to reject non-optimal criteria. Our piece of software supports them, by generating at random a set of movies (cfr. Fig. 4a) or letting them choose an initial set, rearranging it according to a chosen sorting criterion, and applying the greedy procedure (cfr. Fig. 4b). Thus students may experiment different sorting criteria on different instances of the scheduling problem and observe and confront the obtained solutions; they eventually realize that a counterexample is enough to discard a sorting criterion, whereas a proof is needed to accept one as optimal.

Consider for instance the case shown in Fig. 4b. With movies sorted according to decreasing number of overlaps, the greedy procedure would suggest to watch only one movie (the highlighted one on the top), discarding all the remaining ones. However, it would be possible to see four different movies: the sixth (18, 19), the eight (17, 18), the ninth (12, 15), and the tenth (23, 24) from the top, and this is a counterexample proving that the criterion does not guarantee optimality.

Students in different groups tend to find counterexamples for most criteria, mainly observing the outputs produced by the tool on random instances or inventing instances to test a specific idea and detecting the suitable ones. In these cases they should be invited to devise the smallest examples as possible, in order to have insights about the reason why a criterion is not "good in general". However, usually three criteria endure the attempts of finding counterexamples, namely the one based on increasing ending time and its symmetric based on decreasing starting time, and the one considering the number of intersections. Students' attempts then start swinging between building a suitable counterexample and finding an explanation why a counterexample cannot be found for such criteria.

Actually, one can prove that the first one (and its symmetric) guarantees the optimality of the built solution, whereas for the latter counterexamples exist but they are necessarily larger than those for the other criteria. In our experience, no one succeeds



**Fig. 4** The software showing **a** a randomly generated set of movies and **b** applying the greedy procedure according to a selected sorting criterion (number of overlappings)

in getting these results within the time devoted to the activity; however most were eager to know the correct answer and paid strong attention to the formal proof the conductor showed after a while. In several cases, some students asked to delay the final explanation to the next lesson, in order to have the possibility to think further about the problem, and indeed someone succeeded in the task with their own great satisfaction.

### 3.4 Final Recap

In a final recap, the outcome of the previous activity is used to stress that a greedy procedure does not always lead to the optimal solution: this happens for instance if using the non-optimal criteria for the movie festival problem. Moreover, there are some optimization problems that cannot be solved by any greedy algorithm, in that no greedy algorithm is known that can guarantee to find an optimal solution. To let students address this fact on their own, they are asked to reconsider the money change

problem with other sets of coins/bills: is it true that the procedure they wrote at the beginning of the workshop will always output the minimum number of coins/bills, or are there cases when this property is not guaranteed? The question usually surprises students because they had not even contemplated this issue before, but the answer now appears to be not obvious at all. Indeed, after a few attempts, the class is able to devise a suitable set of coins/bills and to make a counterexample. For instance, if we have coins with values 1, 12, 20 and we want to give the change of 24, the greedy algorithm would give 5 coins/bills, namely one piece with value 20 and four pieces with value 1, whereas a better solution would be formed by two pieces with value 12. Other examples of such money systems can be signaled, for instance the British coin system before the decimalisation (15 February 1971) or the one described in Harry Potter's books.

## 4 Conclusion

There is a growing interest in teaching informatics within the standard curricula even in non-vocational school, thus marking a shift with reference to the past identification of this discipline with the use of software applications. Within this trend, we proposed a learning unit rooted on the constructivism and focused on the discovery of greedy strategies in order to solve optimization problems. The learning unit has been developed and fine tuned with bachelor computer science students, and subsequently proposed as an extracurricular activity to some high-school classes between 2015 and 2017. We proposed it mainly as part of a vocational guidance effort and we did not set up any formal assessment. However, the general impression is that most students, even the less mathematically sophisticated, did in fact grasp the idea of an abstract greedy procedure. Many understood the importance to analyze the procedure to check if it works properly for the optimization task. A companion software tool helped several pupils in sensing the impact of the sorting criteria, and realizing that most of the criteria can be discarded through the use of counterexamples.

## References

1. Bellettini, C., Lonati, V., Malchiodi, D., Monga, M., Morpurgo, A., Torelli, M.: Exploring the processing of formatted texts by a kynesthetic approach. In: Proceedings of 7th WiPSCE, pp. 143–144. ACM, NY, USA (2012)
2. Bellettini, C., Lonati, V., Malchiodi, D., Monga, M., Morpurgo, A., Torelli, M.: What you see is what you have in mind: constructing mental models for formatted text processing. In: Proceedings of 6th ISSEP number 6 in *Commentarii informaticae didacticae*, Universitätsverlag Potsdam, pp. 139–147 (2013)
3. Bellettini, C., Lonati, V., Malchiodi, D., Monga, D., Morpurgo, A., Torelli, M., Zecca L.: Extracurricular activities for improving the perception of informatics in secondary schools. In: Gülbahar Y., Karataş E (eds.) Proceedings 7th ISSEP, vol 8730, LNCS, pp. 161–172. Springer (2014)

4. Computing at school: educate, engage, encourage. <http://www.computingschool.org.uk/> (2013)
5. Dagiene, V.: Information technology contests—introduction to computer science in an attractive way. *Inform. Educ.* **5**(1), 37–46 (2006)
6. Henderson, P.B., Cortina, T.J., Wing J.M.: Computational thinking. In: Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education, SIGCSE '07, pp. 195–196. ACM, NY, USA (2007)
7. Hmelo-Silver, C.E.: Problem-based learning: what and how do students learn? *Educ. Psychol. Rev.* **16**(3), 235–266 (2004)
8. International Society for Technology in Education & Computer Science Teachers Association. Operational definition of computational thinking for K-12 education. <https://csta.acm.org/Curriculum/sub/CurrFiles/CompThinkingFlyer.pdf> (2011)
9. Kleinberg, J., Tardos, E.: *Algorithm Design*. Pearson Education India (2006)
10. Kolb, D.A., et al.: Experiential learning theory: previous research and new directions. *Perspect. thinking learn. cogn. styles* **1**, 227–247 (2001)
11. Lonati, V., Malchiodi, D., Monga, M., Morpurgo, A.: Is coding the way to go? In: Brodnik, A., Vahrenhold, J. (eds.) Proceedings of 8th ISSEP. LNCS, vol. 9378, pp. 165–174. Springer, Switzerland (2015)
12. Lonati, V., Malchiodi, D., Monga, M., Morpurgo, A.: Nothing to fear but fear itself: introducing recursion in lower secondary schools. In: Proceedings of the Fifth International Conference on Learning and Teaching in Computing and Engineering (LATICE 2017) (2017) (To appear)
13. Lonati, V., Monga, M., Morpurgo, A., Torelli, M.: What's the fun in informatics? Working to capture children and teachers into the pleasure of computing. In: Kalaš, I., Mittermeir, R.T. (eds.) Proceedings of 4th ISSEP, vol. 7013, LNCS, pp. 213–224. Springer (2011)
14. Partovi, H., Sahami, M.: The hour of code is coming!. *SIGCSE Bull.* **45**(4), 5–5 (2013)
15. Piaget, J., Inhelder, B.: *The Psychology of the Child*. Basic Books, New York (1969)
16. Syslo M.M., Kwiatkowska, A.B.: The challenging face of informatics education in Poland. In: Mittermeir, R.T., Syslo, M.M. (eds.) Proceedings of Informatics Education—Supporting Computational Thinking, Third International Conference on Informatics in Secondary Schools—Evolution and Perspectives, ISSEP 2008, pp. 1–18. Springer (2008)
17. Vygotsky, L.: *Mind in Society: Development of Higher Psychological Processes*. Harvard University Press, Cambridge (1978)

# Comparison of IP and CNF Models for Control of Automated Valet Parking Systems

Abdullah Makkeh and Dirk Oliver Theis

**Abstract** In automated valet parking system, a central computer controls a number of robots which have the capability to move in two directions, under cars, lift a car up, carry it to another parking slot, and drop it. We study the theoretical throughput limitations of these systems: Given a car park layout, an initial configuration of a car park (location of cars, robots), into a desired, terminal configuration, what is the optimal set of control instructions for the robots to reorganize the initial into the terminal configuration. We propose a discretization and compare an Integer Programming model and a CNF-model on real-world and random test data.

**Keywords** Discrete optimization and control · Emerging applications · Logistics

## 1 Introduction

In May 2015, the Miami Herald [7] printed an article about a fancy condominium tower in Miami. That building was equipped with a state-of-the-art automated valet parking system. In these systems, a central computer controls a small army of robots moving in the parking area. The robots have the capability to move in two directions, move under cars, lift a car up, carry it to another parking slot, and drop it. A human driver drops off his car at a *vehicle transfer station*, where the car is picked up by a robot. The robots store away the car in the parking lot until the driver requests to retrieve it. Systems like these are installed in parking lots of apartment complexes, airports, and malls in many countries.

---

A. Makkeh (✉) · D.O. Theis  
Institute of Computer Science of the University of Tartu,  
Ülikooli 17, 51014 Tartu, Estonia  
e-mail: abdullah.makkeh@ut.ee

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_24

233

According to the Miami Herald, in that apartment complex Miami, the following was happening. In the mornings, when the inhabitants of the apartments wanted to drive to work, the system was overloaded: Its algorithms were not able to steer the robots to satisfy a large number of requests within a small time window. Car owners experienced long waiting times (20–30 min) between the time when they requested their cars, and when they were delivered to the vehicle transfer stations. According to the Miami Herald, the car owners took Ubers to work, instead of waiting for the sluggish machines to drag out their cars. The operator promised improvement of the algorithms [7], but ultimately had to shut down the project [6], went into bankruptcy [8], and was sued by the residents [4].

This example illustrates nothing else than the need for optimization in the robot operated car parking systems: according to the New York Times, “the technology is there” [12].

In this paper, we study a fundamental optimization problem involved in the control of an automated car parking system. We study the theoretical throughput limitations of these systems: Given a car park layout, an initial configuration of a car park (location of cars, robots), into a desired, terminal configuration, what is the optimal set of control instructions for the robots to reorganize the initial configuration into the terminal configuration. By “optimal”, we mean fastest, in terms of clock-on-the-wall waiting time until the robots have completed their tasks.

With the ultimate goal of heuristic algorithms for the control of the robots, in this paper, we study exact algorithms. We propose an Integer Programming (IP) model and a Constraint Programming model (Boolean variables with constraints in conjunctive normal form, CNF), and compare the two approaches by testing them on parking lot scenarios.

In an attempt to stay very close to the application, both approaches model very closely a realistic model of the way a particular type of robots operates, with specific velocities of loaded and empty robots along with acceleration times, speeds of car lift and drop, etc. The goal of the engineers who designed these robots was to turn existing, human-operated parking lots into robot operated ones, rather than building the whole parking lot from scratch, with the idea to increase the packing density of cars on parking lots in densely populated cities. In particular, in this paper we will consider the case study of parking lots in big condos in Tallinn, Estonia.

The rest of this paper is organized as follows. In the next section, we review literature relevant for our problem. In Sect. 3, we give an overview of the IP model and the CNF clauses. In Sect. 4 we describe the data, the computational experiments we ran, and we discuss the obtained results. In Sect. 5, we draw some conclusions regarding the problem.

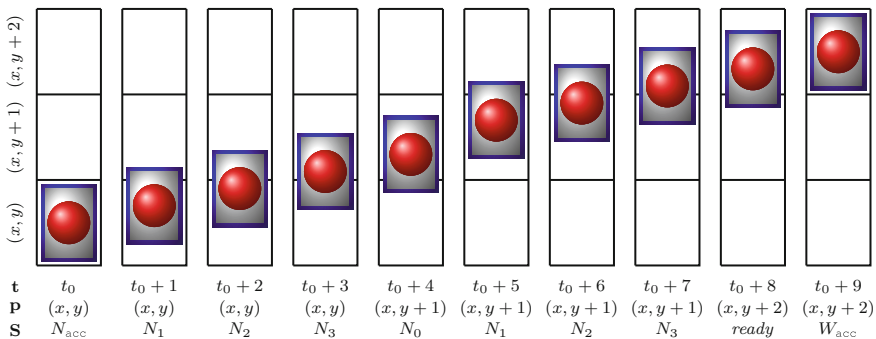
We defer to the supplement [5] for full details of our models, in case the referees would like to verify their soundness.

## 2 Basic Approach and Related Work

### 2.1 Problem Data and Basic Modelling Approach

The technical details of the problem were the result of a collaboration with a company which considered entering the market of automated valet parking installations. In the considered parking lots, all slots have the same width (3 m) and length (6 m), and all slots are parallel: the width is in East-West direction, the length in North-South direction. The parking lots have a rectangular bounding box: e.g., if the bounding box is 300 m in wide and 600 m long, the parking lot can contain up to  $10 \times 10 = 100$  slots. Correspondingly, slots are identified by  $x$  (East-to-West) and  $y$  (South-to-North) coordinates. We allow for slots to be unusable, due to either obstacles (walls, pillars, broken down robots) or simply parked cars which, for some reason, we currently don't want to move. The robots can move either in North-South or in East-West direction (but not, e.g., diagonally); also they don't need to "rotate". A robot must come to a complete stand still before changing directions. The robots' maximum speed is 3 m/s when empty, or 1.5 m/s when carrying a car; they require 1 s to accelerate to maximum speed or decelerate from maximum speed to standing still. (We don't allow for a robot to stand still between two slots.) The robot needs 6 s to lift a car and 2 s to drop it.

These data suggest to discretize time into intervals of 0.25 s. E.g., Fig. 1 shows a complete sequence of a North-movement by a robot carrying a car from a slot  $(x, y)$  to a slot  $(x, y + 2)$  and then accelerate West. In time interval  $t_0$ , the robot picks up speed, and moves 1/8 of the length of a slot. In time intervals  $t_0 + 1, \dots, t_0 + 7$ , the robot moves at full speed. In time interval  $t_0 + 8$ , it slows down and comes to a stand 2 places north of where it started. The control instruction "ready" causes the robot to initiate stopping. In time interval  $t_0 + 9$ , the robot could be executing a new control instruction; in the figure, it is accelerating West.



**Fig. 1** North movement of a loaded robot. **t** = time interval; **p** = position associated and **S** = stage of the move. Robots is drawn in the physical location which it occupies at the beginning of the time interval



The discrete optimal control problem we aim to solve is now the following: Given two configurations of locations of robots and cars on the parking lot, an “initial configuration” and a “terminal configuration”, determine a feasible set of control instructions for the robots which transforms the initial configuration into the terminal configuration; minimize the time the reconfiguration takes. Clearly equivalent to a set of control instructions is a sequence of configurations, each one of which can be derived from the previous one by a feasible move of the robot.

In a practical application, there is usually no need to fix initial and terminal locations of each individual car. It is, e.g., not relevant whether car #47 goes to vehicle transfer station #23 and car number #54 goes to vehicle transfer station #22—or vice versa. Hence, we work with “colored” cars, and the initial and terminal configurations specify only whether a slot is occupied by a car and if so, what the “color” of that car should be. In our computer code, we formulate and solve IP and CNF models whose feasible solutions are sequences of configurations, for 3 colors.

## 2.2 Simplifications and Literature Review

Simplified versions of the problem have been studied theoretically. Most importantly, disregarding the role of robots (i.e., assuming that the cars move by themselves) and the speeds gives variants of pebble motion and reconfiguration problems in grids, or, more generally graphs: Vertices represent parking slots, and edges represent slots sharing an edge. We now review what is known.

The most famous problem in this group is the *15-Puzzle*: on a  $4 \times 4$  grid, 15 vertices are occupied by labeled pebbles. (“Labeled” means that each pebble has a color of its own.) The decision version of the problem is whether a given initial configuration can be transformed into a given terminal configuration through a sequence of moving pebbles to adjacent vertices; the optimization version asks for a reconfiguration with the smallest number of total moves. The decision problem can be solved in polynomial time [14]. The optimization problem on grids is NP-hard [11], but constant factor approximation algorithms exist [10].

For graphs in general, Papadimitriou et al. [9] proved that with two labels, one pebble has a unique label, the problem is NP-hard. They also gave a polynomial time algorithm, using flow techniques, for the optimal solution on trees. The result was later improved to  $O(n^5)$  by Auletta et al. [1]. Călinescu et al. [2] showed that the optimization problem on graphs is APX-hard. Moreover, when allowing more pebbles to move at the same time, Yu and LaValle [15] proved that the optimization problem on graphs is NP-hard. It follows from [2, 15] that there is no polynomial time (approximation) scheme for the simplified version of our problem, unless  $P = NP$ .

### 3 IP and CNF Models

Let  $C := \{0, 1, \dots, |C| - 1\}$  denote the set of colors of cars. The models require an upper bound on the number of time intervals: Each time interval is identified by an element of  $T = \{0, 1, \dots, t_{\max}\}$ . The configuration at  $t = 0$  is the initial configuration, the configuration at  $t = t_{\max}$  is the terminal configuration. A car is called *stationary* if it is not carried by a robot.

#### The Variables

For every time interval and every slot, four types of variables determine what is happening there and then: *slot occupation status* (whether there's a stationary car, and if so, of what color), *slot robot-status, SRS* (whether there's a robot, and if so, what it is carrying and doing), *robot vertical process* (if there is a robot lifting or dropping a car, which phase of that process is it executing), *robot horizontal movement* (if there is a moving robot, which phase of the movement is it executing). All variables are Boolean (0/1). At most one of the variables in each group is 1. Here they are.

*Slot Occupation Status*  $x_{v,t,c}$ , for every slot  $v = (x, y)$ , every time interval  $t = 0, \dots, t_{\max}$ , and every  $c \in \{\phi\} \cup \{C\}$ . The symbol  $\phi$  stands for “no car”. Example:  $x_{v,t,\phi} = 1$  iff during time interval  $t$ , there is no stationary car at the slot with coordinates  $v$ ;  $x_{v,t,2} = 1$  iff during time interval  $t$ , there is a stationary car of color 1 at the slot with coordinates  $v$ .

*Slot Robot-Status (SRS)*  $s_{v,t,c,d}$ , for every slot  $v$ , every time interval  $t$ , every  $c \in \{\phi\} \cup \{C\}$ , and every  $d \in \{\epsilon, \rho, \mu, \zeta\}$ . The symbol  $\epsilon$  stands for “no robot”,  $\rho$  stands for “ready”,  $\mu$  stands for “robot moving”,  $\zeta$  stands for “vertical process”. Example:  $s_{v,t,\phi,\rho} = 1$  iff at the slot with coordinates  $v$ , during time interval  $t$ , there is a robot doing executing the “ready” control instruction, i.e., it is doing one of the following: (1) nothing (being idle); (2) ending a movement (decelerating); (3) ending lifting or dropping a car. Another example:  $s_{v,t,2,\zeta} = 1$ , iff at the slot with coordinates  $v$ , during time interval  $t$ , there which is in the process of either lifting or dropping a car of type 2.

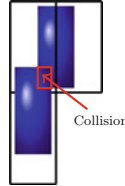
*Robot Vertical Process (RVert)*  $z_{v,t,p}$ , for every slot  $v$ , every time interval  $t$ , and every  $p \in \{\lambda_0, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \delta\}$ . The symbols  $\lambda_i$  for  $0 \leq i \leq 4$  stand for the phases of “lifting”,  $\delta$  stands for “dropping”. Example:  $z_{v,t,\lambda_2} = 1$  iff during time interval  $t$ , the robot is in the third phase of lifting at the slot with coordinates  $v$ . The following diagram shows the interconnection between  $x$ -,  $s$ -, and  $z$ -variables:

$t = t_0$	$x_{v,t,3} = 1, s_{v,t,\phi,\rho} = 1, z_{v,t,*} = 0$	a car of color 3, a robot
$t = t_0 + 1$	$x_{v,t,\phi} = 1, s_{v,t,3,\zeta} = 1, z_{v,t,\lambda_0} = 1$	no car; a robot lifting a car of color 3
$\vdots$	$\vdots$	$\vdots$
$t = t_0 + 5$	$x_{v,t,\phi} = 1, s_{v,t,3,\zeta} = 1, z_{v,t,\lambda_4} = 1$	no car; a robot lifting a car of color 3
$t = t_0 + 6$	$x_{v,t,\phi} = 1, s_{v,t,3,\rho} = 1, z_{v,t,*} = 0$	no car; robot executing “ready”

*Robot Horizontal movements (RMove)*  $m_{v,t,c,e}$ , for every slot  $v$ , every time interval  $t$ , every  $c \in \{\phi\} \cup \{C\}$ , and every  $e \in \{N_{acc}\} \cup \{N_{i,z} \mid 0 \leq i \leq 3\} \cup \{S_{acc}\} \cup \{S_{i,z} \mid 0 \leq i \leq 3\}$

$i \leq 3 \} \cup \{E_a\} \cup \{E_i \mid 0 \leq i \leq 1\} \cup \{W_{acc}\} \cup \{W_i \mid 0 \leq i \leq 1\}$ . The symbol  $N_{acc}$  stands for “acceleration in the north direction” and  $N_i$  for  $0 \leq i \leq 3$  stands for the phases of “northern move”. Similarly the other stand for the other directions. Example:  $m_{v,t,2,E_1} = 1$  iff during time interval  $t$ , the robot moving  $E_1$  at the slot with coordinates  $v$  towards its east. The interconnection between the  $x$ - and the  $m$ -variables is similar to the RVert case.

**IP-Model Constraints**



The setup of the variables allow to formulate both the IP- and the SAT-model by linking only consecutive time intervals  $t, t + 1$ . Care has to be taken to allow for all feasible movements, while making sure that collisions (see the figure on the right) are ruled out. The following constraints describe a loaded robot on  $(x, y)$  moving north.

$$\sum_{i=0}^{|C|-1} m_{(x,y),t,i,N_0} \leq s_{(x,y+1),t,\phi,\epsilon} + \sum_{i=0}^{|C|-1} (m_{(x,y+1),t,i,N_0} + m_{(x,y+1),t,i,N_{acc}} + m_{(x,y+1),t,i,1} + m_{(x,y+1),t,i,N_2} + m_{(x,y+1),t,i,N_3}) + m_{(x,y+1),t,\phi,N_{acc}} + m_{(x,y+1),t,\phi,N_1} \tag{1}$$

$$\sum_{i=0}^{|C|-1} m_{(x,y),t,i,N_{acc}} \leq s_{(x,y+1),t,\phi,\epsilon} + \sum_{i=0}^{|C|-1} (m_{(x,y+1),t,i,N_{acc}} + m_{(x,y+1),t,i,N_1} + m_{(x,y+1),t,i,N_2} + m_{(x,y+1),t,i,N_3}) + m_{(x,y+1),t,\phi,N_{acc}} + m_{(x,y+1),t,\phi,N_1} \tag{2}$$

$$\sum_{i=0}^{|C|-1} m_{(x,y),t,i,N_1} \leq s_{(x,y+1),t,\phi,\epsilon} + \sum_{i=0}^{|C|-1} (m_{(x,y+1),t,i,N_1} + m_{(x,y+1),t,i,N_2} + m_{(x,y+1),t,i,N_3}) + m_{(x,y+1),t,\phi,N_1} \tag{3}$$

$$\sum_{i=0}^{|C|-1} m_{(x,y),t,i,N_2} \leq s_{(x,y+1),t,\phi,\epsilon} + \sum_{i=0}^{|C|-1} (m_{(x,y+1),t,i,N_2} + m_{(x,y+1),t,i,N_3}) \tag{4}$$

$$\sum_{i=0}^{|C|-1} m_{(x,y),t,i,N_3} \leq s_{(x,y+1),t,\phi,\epsilon} + \sum_{i=0}^{|C|-1} m_{(x,y+1),t,i,N_3} \tag{5}$$

$$\sum_{i=0}^{|C|-1} m_{(x,y),t,i,N_0} \leq s_{(x,y-1),t,\phi,\epsilon} + \sum_{i=0}^{|C|-1} m_{(x,y-1),t,i,N_0} \quad (6)$$

First (1) says that if a robot is moving in stage  $N0$  then *north* of it there can be nothing or robot accelerating in the same direction as it and so on. E.g., when the robot is moving  $N1$  from (3) there can't be a robot to the *north* of it accelerating since a collision will occur. Moreover, since  $N0$  is the stage of north movement when the robot arrives to  $(x, y)$  and is almost occupying all of this spot, then by (6)  $(x, y - 1)$  is unoccupied or there is a robot moving  $N0$ . We defer to the supplement [5] for the full details of the IP-model. Now we state the following result.

**Proposition 1** *The IP-formulation using the  $x$ -,  $s$ -,  $z$ -, and  $m$ -variables on an  $n \times n$  grid has  $O(t_{\max} n^2)$  inequalities.*

**The Objective Function.** We made numerous experiments with different objective functions. The idea is to “nudge” the robots into movement, while trying to support reaching the terminal configuration as quickly as possible. For each time interval,  $f(t) = \frac{t}{100 \cdot t_{\max}}$  was the cost of the  $SRS$  excluding  $s_{v,t,\phi,\rho}$  and  $s_{v,t,\phi,\epsilon}$ . The objective function is as follows,

$$\sum_{\forall v \in V, \forall t \in T, \forall i \in C} f(t) \cdot (s_{v,t,i,\rho} + s_{v,t,\phi,\mu} + s_{v,t,i,\mu} + s_{v,t,\phi,\epsilon}) \quad (7)$$

The computational results in the next section use a linear dependence on the time for that. Details can be found in the supplement [5].

### SAT-Model Clauses

The SAT-model follows the same basic approach as the IP-model. Indeed, many of the clauses have a direct counterpart in an inequality, and vice versa. Overall, deriving the CNF clauses from the control requirements and the variables is an exercise in logical thinking. The following clauses determine the situation of a slot  $v$  at time  $t$  after lifting or before dropping.

$$\forall i \in C \ s_{v,t+1,i,\rho} \vee_{\substack{j \in C \\ j \neq i}} s_{v,t+1,j,\rho} \vee x_{v,t,\phi} \vee z_{v,t,\lambda_4} \quad (8)$$

$$\forall i \in C \ s_{v,t,i,\rho} \vee_{\substack{j \in C \\ j \neq i}} s_{v,t,j,\rho} \vee x_{v,t,\phi} \vee z_{v,t+1,\delta} \quad (9)$$

$$\forall i, j \in C \ \neg s_{v,t+1,i,\rho} \vee \neg s_{v,t+1,j,\rho} \quad (10)$$

$$\forall i, j \in C \ \neg s_{v,t,i,\rho} \vee \neg s_{v,t,j,\rho} \quad (11)$$

By (8) and (10), in  $t + 1$ , if a loaded robot (car of type  $i$ ) will be ready on  $v$ , then, in  $t$ ,  $v$  has no cars or the robot is done with lifting a car of type  $i$ . Similarly (9) and (11), if in  $t$ , the loaded robot is ready on  $v$ , then, in  $t + 1$ ,  $v$  will have no cars or the robot will drop a car of type  $i$ . We defer to the supplement [5] for the full details.

## 4 Computational Results

**The data.** We have two sets of parking lots. Five parking lots are derived from an existing parking area in Tallinn, Estonia. That parking area has been split up into parts, based on requirements such as having vehicle transfer stations accessible from elevators for pedestrians. (Humans and robots cannot use the same floor area for operational safety.) Then there is a set of 5 parking lots generated randomly: Starting from an  $m \times n$  grid, we place “walls” between parking slots with a given probability.

Each parking lot gives rise to several instances to be solved, by choosing the initial and terminal configurations of the cars and robots. All configurations were chosen randomly (discarding those with no reconfiguration of initial to terminal configurations.) The total number of instances is 30. The largest instances were  $13 \times 10$  (only 41 free spot). In those 7 cars of 3 different colors and 5 robots were used. Also we had small instances ( $4 \times 5$  and 9 free spots) with 6 cars of 3 different colors and 2 robots. Compared to real world problems, the instances are small in terms of number of cars, but the sizes and number of colors are compatible with real world ones. The full description of the all parking lots and instances can be found in the supplement [5].

Finally each instances is solved several times, for varying  $t_{\max}$ . (The difficulty is that no good upper bound on  $t_{\max}$  is known.)

**Hard- and Software** We used the Gurobi optimizer [3] in version 7.0 for the IP-model, and CryptoMiniSat 5 [13] as SAT solver. The times were taken on a compute server with Intel(R) Core(TM) i7-4790K CPU (4 cores) and 16 GB of RAM. (Both solvers were run with one thread.) Gurobi was run with the parameters  $MIPFocus = 2$ ,  $Presolve = 2$ . CryptoMiniSat was run with  $preproc = 1$ . Both solvers were given a time limit of 10000 s.

The time to read in the data from file and construct the models was negligible.

**Results.** CryptoMiniSat 5 very significantly outperformed Gurobi.

For only one of the resulting Integer Programs a solution was found within the time limit. For that one IP, in that case, the solve time was 90 s, whereas the solve time for the corresponding CNF model was 76 s.

The SAT-model, on the other hand, gave relatively satisfactory results. CryptoMiniSat terminated within the given time limit for 85% the CNF-models (yielding either a feasible solution or the information that there was none). In 60% of instances, the running time was below 3600 s. In only 20% instances, the running time was larger than 6000 s.

The full computational results (tables) can be found in the supplement [5]. The code, instances and solutions are on GitHub ([Abzinger/crobots](https://github.com/Abzinger/crobots)).

## 5 Conclusion

Difficult discrete optimal control problems arise in the control of automated valet parking systems. For the study of exact methods for the problems, we devised a time-expanded model, and compared its solution via Integer Programming and CNF-based Constraint Programming. Using state of the art software for the two approaches, we found that Integer Programming was useless.

The CNF-model, however, proved to be usable. Now, the next goal is to (design and) compare the solution qualities (amount of time needed to reach a terminal configuration from a current configuration) of heuristic algorithms to optimal solutions, which can be found or at least bounded using our CNF-model.

**Acknowledgements** Supported by the Estonian Research Council, ETAG (*Eesti Teadusagentuur*), through PUT Exploratory Grant #620, and by the European Regional Development Fund through the Estonian Center of Excellence in Computer Science, EXCS.

## References

1. Auletta, V., Monti, A., Parente, M., Persiano, P.: A linear-time algorithm for the feasibility of pebble motion on trees. *Algorithmica* **23**(3), 223–245 (1999)
2. Călinescu, G., Dumitrescu, A., Pach, J.: Reconfigurations in graphs and grids. *SIAM J. Discret. Math.* **22**(1), 124–138 (2008)
3. Gurobi Optimization, Inc.: Gurobi optimizer reference manual (2016)
4. Lima, D: Brickell condo residents sue developer over faulty robotic garage. *Miami Herald*, 20 April 2016
5. Makkeh, A., Theis, D.O.: Comparison of IP and CNF models for control of automated valet parking systems. Supplementary material, University of Tartu (2017). <http://ac.cs.ut.ee/files/robot-supplement.pdf>
6. Nehamas, N.: Residents furious as robotic parking garage at Brickell condo shuts down. *Miami Herald*, 6 Nov 2015
7. Nehamas, N.: Robotic parking garage ruins commute, Brickell condo residents say. *Miami Herald*, 15 May 2015
8. Nehamas, N.: Operator pulls out of Brickell condos disastrous robotic parking garage. *Miami Herald*, 11 Jan 2016
9. Papadimitriou, C.H., Raghavan, P., Sudan, M., Tamaki, H.: Motion planning on a graph. In: *Proceedings of the 35th Annual Symposium on Foundations of Computer Science 1994*, pp. 511–520. IEEE (1994)
10. Ratner, D., Warmuth, M.: The  $(n-1)$ -puzzle and related relocation problems. *J. Symb. Comput.* **10**(2), 111–137 (1990)
11. Ratner, D., Warmuth, M.K.: Finding a shortest solution for the  $n \times n$  extension of the 15-puzzle is intractable. In: *AAAI*, pp. 168–172 (1986)
12. Robles, F.: Road to robotic parking is littered with faulty projects. *The New York Times*, 27 Nov 2015
13. Soos, M.: The CryptoMiniSat 5 set of solvers at sat competition 2016. In: *SAT Competition 2016*, p. 28 (2016)
14. Wilson, R.M.: Graph puzzles, homotopy, and the alternating group. *J. Comb. Theor. Ser. B* **16**(1), 86–96 (1974)
15. Yu, J., LaValle, S.M.: Structure and intractability of optimal multi-robot path planning on graphs. In: *AAAI* (2013)

**Part VI**  
**Location**

# A Districting Model to Support the Redesign Process of Italian Provinces

Giuseppe Bruno, Antonio Diglio, Alessia Melisi and Carmela Piccolo

**Abstract** In the general context of welfare reforms in western economies, many actions concerning the rationalization of local administrative structures have been undertaken. In particular, in Italy a recent debate has been addressed about the reduction of the overall number of provinces and the rearrangement of their borders. As provinces are responsible of providing some essential services to the population within their boundaries, any possible scenario should combine the need for more efficient territorial configurations with the safeguard of the services' accessibility. From a methodological point of view, such problem involves aspects from both facility location and districting problems. In this work, we formulate a mathematical model to support the decision making process and we compare scenarios provided on four benchmark problems, built on the real data associated to the most representative Italian regions.

**Keywords** Districting problems • Facility-location models • Territorial re-organization

---

G. Bruno · A. Diglio (✉) · A. Melisi · C. Piccolo  
Department of Industrial Engineering, University of Naples Federico II,  
P.le Tecchio 80, Naples, Italy  
e-mail: antonio.diglio@unina.it

G. Bruno  
e-mail: giuseppe.bruno@unina.it

A. Melisi  
e-mail: alessia.melisi@unina.it

C. Piccolo  
e-mail: carmela.piccolo@unina.it



## 1 Introduction

From an administrative point of view, the Italian country is subdivided into *regions* (NUTS2), *provinces* (NUTS3) and *municipalities* (LAU) [1]. Each government level is characterized by specific competences and it is responsible of providing some essential services to the population living within the borders of the related areas. Chapter 5 of Italy's Constitution focuses on the way power should be distributed and divided between the central government and various local administrative levels.

Recently, a heated debate has concerned the opportunity of reducing the overall number of provinces in the country and of redesigning their borders. Such proposal has mainly twofold motivations. On one side, the economic crisis imposed the need of reducing public expenditure through the reorganization of systems providing public services, such as healthcare, education, justice. In this context, the redesign of provinces would allow the reduction of management costs through the amalgamation of political jurisdictions and their public service provision areas. On the other side, in the last years, the competences traditionally assigned to provinces have been gradually reduced and, at the current state, they take care solely of road network management and education, with reference to the upper secondary level [2]. For these reasons, the actual configuration, characterized by a huge number of provinces and an unbalanced distribution of population and area among them, have been considered not sustainable anymore.

The debate around the reorder of Italian Provinces is still ongoing and it is testified by a series of reform proposals, that found huge obstacles in reaching a consensus [3, 4]. Probably, this is due to the difficulty, faced by government, in implementing solutions capable of combining the need for efficient territorial configurations and the safeguard of users' accessibility to the provided services. One of the last governmental proposal provided specific guidelines to support the redesign process, by fixing constraints about the minimum population ( $P_{min} = 300.000$  inhabitants) and the minimum extension ( $S_{min} = 2.500$  km<sup>2</sup>), that all provinces in the final configuration need to meet, and left to the single regions the possibility to propose their own reorder plans, according to the specific exigencies and characteristics [3]. At this aim, the availability of models and methods able to support regional authorities in such complex decision making process could be beneficial.

From a methodological point of view, the described problem involves aspects from both facility location [5, 6] and districting problems [7, 8] (FLP and DP). Indeed, it concerns decisions about facilities (province centers) to be closed in a given study region and, at the same time, about the new partition of municipalities in districts.

In the last years, in the literature related to FLPs, several works started to analyze problems aimed at modifying the territorial configuration of existing facilities in a study region, by closing, relocating, merging some of them and/or by downsizing/redistributing their operating capacities [9–12]. Such *territorial re-organization problems* are usually motivated by occurred or potential changes in the

facilities' operating conditions (i.e., demand distribution and/or budget constraints), that make the existing organization inefficient and/or unsustainable. ReVelle et al. [9] introduce two different models to deal with facilities' closure, both in a competitive and non-competitive environment. Bruno et al. [10–12] focus on applications related to the public context, by considering other reorganization actions, as facilities merging and capacities downsizing. In the public sector, such decisions have to be made by carefully evaluating their potential impact on users' accessibility to services and equity consideration [13, 14]. In particular, Bruno et al. [11] also propose a classification of re-organization models, in the attempt of systematizing the literature focused on this class of problems. In particular, they distinguish between *single-period* and *multi-period* models (see for instance [9, 15]), and between *ex-ante* and *ex-post* models, if decisions are taken before or after changes motivating re-organization occur (see for instance [10, 16]).

In the DP literature, some papers address the problem of modifying the current partition of territorial units in districts [17, 18]. In particular, Bruno et al. [18] introduce four different models to perform re-districting decisions, based on different strategies to take closure (*optimal* vs. *prescriptive*) and units' reallocation decisions (*merging* vs. *re-assignment*).

In this work, with reference to the above literature streams, we formulate a model to support the redesigning process of Italian provinces, that may be viewed as an extension of the Optimal Reassigning Model (ORM) by Bruno et al. [18]. The proposed model is tested on four instances, built on the real data of the most representative Italian regions. Provided results are reported and analyzed, in order to show the usability of the model and its efficacy in providing interesting insights to support the decision making process.

## 2 A Mathematical Model for the Re-design of Italian Provinces

In the current Italian administrative subdivision, each region is composed of a certain number of municipalities  $i \in I$ , grouped in provinces  $j \in J$ . Then, the set  $I$  is partitioned in  $m$  components  $I_j$ , each representing the subset of territorial units assigned to province  $j \in J$  ( $\cup_{j \in J} I_j = I$  and  $I_j \cap I_{j'} = \emptyset \forall j, j' \in J$ ). In each province, a specific municipality  $c_j$  (named *provincial chief town*) hosts facilities providing a set of services to the population living within its borders.

The problem consists of reducing the overall number of active provinces/chief towns and then modifying the partition of municipalities, in order to produce a trade-off solution between the need of obtaining more efficient territorial configuration and the goal of containing as much as possible the impact on the users, in terms of accessibility to the provided services.

In order to take into account the efficiency of the produced scenario, we consider the requirements defined by the central government, in terms of minimum

population  $P_{min}$  (300.000 inhabitants) and area  $S_{min}$  (2.500 km<sup>2</sup>) per province. On the other side, in order to take into account the accessibility of reallocated users/municipalities to the new assigned chief-units, we consider their related distances.

Then, the model is aimed at determining how many and which provinces have to be closed, in such a way the reassignment of municipalities to active provinces will produce feasible configurations (meeting the governmental requirements) and will optimize the accessibility to services provided by new province centers.

By introducing the binary variables  $y_j$ , equal to 1 if and only if the chief town of district  $j$  gets closed, and  $x_{ij}$ , equal to 1 if and only if unit  $i$  is assigned to the chief town of district  $j$ , the model can be formulated as follows:

$$\min z = \sum_{i \in I} \sum_{j \in J} p_i d_{ic_j}^2 x_{ij} \quad (1)$$

$$x_{ij} \leq 1 - y_j \quad \forall i \in I, \forall j \in J \quad (2)$$

$$\sum_{j \in J} x_{ij} = 1 \quad \forall i \in I \quad (3)$$

$$\sum_{i \in I_j} x_{ij} \geq \gamma |I_j| (1 - y_j) \quad \forall j \in J \quad (4)$$

$$\sum_{i \in I} p_i x_{ij} \geq P_{min} (1 - y_j) \quad \forall j \in J \quad (5)$$

$$\sum_{i \in I} s_i x_{ij} \geq S_{min} (1 - y_j) \quad \forall j \in J \quad (6)$$

$$y_j \in \{0, 1\}; x_{ij} \in \{0, 1\} \quad \forall i \in I, j \in J \quad (7)$$

The objective function (1) is one of the classical measures of compactness, defined as sum of the weighted square distances among each unit  $i$  and the chief town of its assigned district  $j$ . Constraints (2–4) rule the reallocation of territorial units to active districts. In particular, constraints (2) impose that territorial units can be assigned only to active districts, while constraints (3) ensure the allocation of each territorial unit to one (and only one) district. Constraints (4) impose a similarity condition for active provinces; in particular, they impose, for each active province, that at least a given percentage  $\gamma$  ( $0 \leq \gamma \leq 1$ ) of units initially assigned to  $j$  ( $i \in I_j$ ) have to be assigned to  $j$  in the final configuration. For  $\gamma = 1$ , units belonging to active districts cannot be reassigned, while for  $\gamma = 0$ , any unit may be re-allocated and the initial partition does not affect the final solution. Conditions (5–6) represent the requirements constraints. In particular, they impose that active districts in the final configuration will be characterized by an area larger than  $S_{min}$  and a population larger than  $P_{min}$ . The total area and population for each district  $j$  have been computed by summing up the population  $p_i$  and the area  $s_i$  of the single municipalities assigned to it. Finally, constraints (7) define the nature of the decision variables.

### 3 Results

The introduced model has been tested on the most representative Italian Regions, in terms of number of districts and municipalities. The selected regions are indicated in Table 1, along with some features associated to their global characteristics, in terms of total population and area, and others associated to their current partition in provinces, such as the minimum, maximum and average values of population, area and radius per province (i.e. the distance between the province chief town and its farthest municipality). It is possible to notice that, in all the cases, the distribution of population and area across provinces is not balanced, being the range between min and max values very significant. Moreover, the current partition in provinces do not satisfy the government requirements, being the minimum values under the fixed thresholds.

In order to solve the model, for each considered region, population and extension associated to each municipality have been retrieved by the National Office of Statistics; while the distances have been calculated as shortest paths (km) on the road network. The test problems have been solved using Cplex 12.2 on an Intel Core i7 with 1.86 GHz and 4 GB of RAM, in very limited running times.

The model has been solved first assuming  $k$  as decision variable, so as to find the minimum number of chief towns ( $k_{min}$ ) to be closed in order to have feasible provinces, and then by considering it as a parameter and varying for higher values, (i.e. by introducing in the model the additional constraint  $\sum_{j \in J} y_j = k, k > k_{min}$ ).

In order to compare different scenarios, in the following we report solutions with the same number of active provinces, equal to 5, obtained by fixing a similarity value equal to 1. In particular, for each considered region, the map representing the scenario proposed by the model (Figs. 1, 2, 3 and 4), along with a table (Tables 2, 3, 4 and 5) indicating its characteristics, in terms of number of territorial units, total population, total extension (km<sup>2</sup>) and radius per province, are reported.

It is possible to notice that new provinces are feasible, being their total population and extension higher than the governmental requirements. However, the distribution among provinces is quite different within the four Regions. In particular, in the case of Piemonte Region, the population is not well balanced, as all the values are under the threshold of 610.000 inhabitants except one that is higher than 2.000.000. The other regions present a better population distribution, especially Veneto and Emilia Romagna. In terms of area, Toscana Region does not present a good balance, being the minimum value equal 2.585,7 km<sup>2</sup> and the maximum one equal to 6.440,5 km<sup>2</sup>. In terms of provincial radius, Piemonte region presents the worst condition (154,8 km), while Veneto Region present the more balanced distribution, being all the provincial radius within the range 75,0–95,0 km.

In order to evaluate how varies the accessibility of users to provincial chief units when the number of closed provinces increases, we report, for each region, the results provided by the model, in terms of minimum, maximum and average provincial radius, by varying  $k$  (Fig. 5). In each Figure, also the value associated to the current partition (CP) is reported.






**Table 1** Characteristics of the selected regions

$m$	Total pop. ( $10^3$ )	Total area ( $\text{km}^2$ )	Provincial population ( $10^3$ )			Provincial area ( $\text{km}^2$ )			Provincial radius (km)		
			Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
Toscana	3.672,2	22.987	378,4	199,7	973,1	2.130,0	365,7	4.503,1	81,8	29,9	148,2
Piemonte	4.356,9	25.377	544,6	160,3	2.240,8	3.172,2	913,3	6.894,9	80,9	36,8	114,4
Veneto	4.857,2	18.407	693,9	210,0	921,4	2.629,6	1.819,4	3.672,3	70,9	53,1	93,6
E.Romagna	4.342,1	22.453	482,5	284,6	976,2	2.494,8	864,9	3.702,3	73,9	55,3	90,1

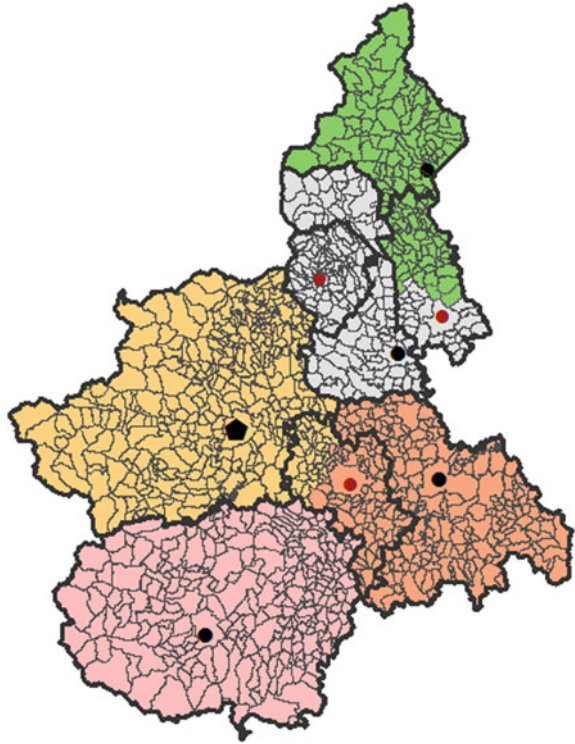
**Fig. 1** Toscana (k = 5)








**Table 2** Toscana - New provinces characteristics

	TUs	Pop	Area (kmq)	R <sub>mj</sub> (km)
	49	368.534	2.585,7	112,4
	74	1.585.999	5.477,3	75,2
	56	925.559	3.604,8	112,4
	56	434.974	4.878,8	89,4
	52	357.136	6.440,5	122,6

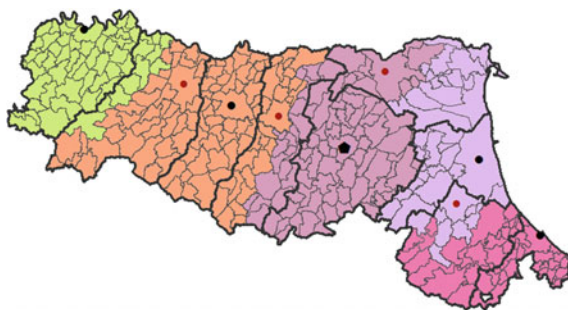
As expected, the accessibility gets worse when the number of closed provinces increases. However, this representation may be very useful as it allows to compare different scenarios according to the two conflicting objectives to be taken into account in the decision-making process (i.e. efficiency vs. accessibility). For example, in the case of Toscana Region, it is interesting to see that by increasing the number of closed provinces from 5 to 6, the situation does not change too

**Fig. 2** Piemonte (k = 3)**Table 3** Piemonte - New provinces characteristics






	TUs	Pop	Area (kmq)	R <sub>mj</sub> (km)
	141	355.109	3.085,0	154,8
	347	2.275.353	7.167,4	99,6
	197	534.011	3.577,7	105,5
	250	586.378	6.894,9	114,4
	271	606.053	4.652,4	70,7

much; indeed, the average is quite constant as well as the range between min and max. In this condition, the decision maker could decide to close 6 instead of 5 provinces without producing a significant worsening in the users' conditions. On the contrary, for values higher than 6 the radius increases significantly. In the case of Emilia Romagna Region, different scenarios are characterized by similar radius,

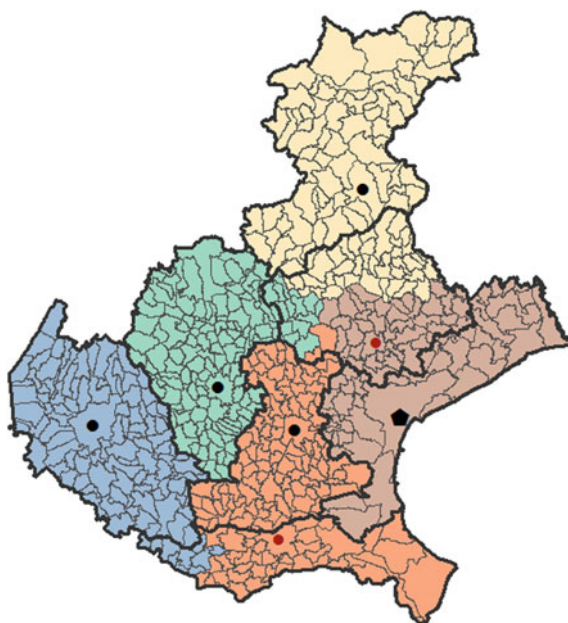
**Fig. 3** Emilia Romagna  
( $k = 4$ )



**Table 4** Emilia Romagna -  
New provinces characteristics






	TUs	Pop	Area (kmq)	$R_{mj}$ (km)
	59	359.139	3.342,4	77,6
	107	1.348.923	6.538,2	117,1
	92	1.419.964	5.915,6	82,3
	41	672.089	4.097,0	81,9
	49	542.020	2.559,7	106,5

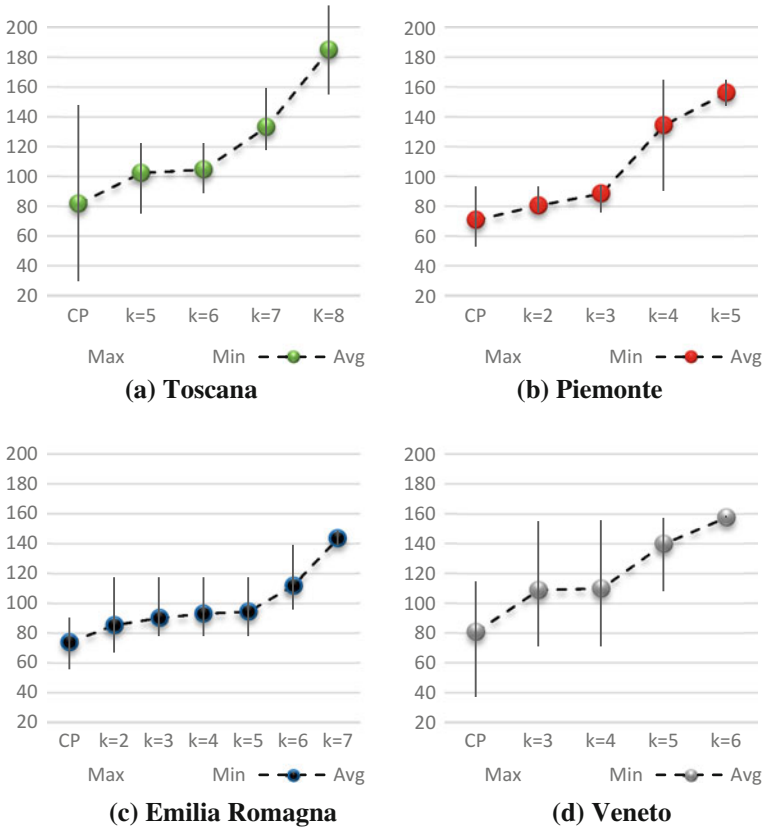
**Fig. 4** Veneto ( $k = 2$ )





**Table 5** Veneto - New provinces characteristics

	TUs	Pop	Area (kmq)	R <sub>mj</sub> (km)
	108	922.251	3.303,2	78,0
	135	976.036	3.045,3	75,9
	107	475.074	4.621,7	76,0
	85	1.316.124	3.593,8	93,6
	146	1.167.725	3.843,4	79,5



**Fig. 5** Provincial radius in the produced scenarios

so it could be interesting for the decision maker comparing the characteristics of single scenarios into details, in order to evaluate if it is reasonable to enforce the closure of an higher number of provinces without damaging too much users.

## 4 Conclusions

In this work, we introduced a model to address the real problem of the reorder of Italian provinces and we tested it on some benchmark problems, built on real-world instances derived from Italian regions territorial configurations.

Moreover, a sensitivity analysis on the number of closed provinces has been performed, in order to evaluate how the accessibility of users to provincial chief units varies when the number of closed provinces increases. This analysis is very useful in order to compare scenarios on the basis of the two conflicting goals to be considered in the decision-making process.

Finally, computational results highlight that models can be solved in very limited running times, even for instances of significant size.

Further researches will be addressed at enhancing the models formulations in order to take into account further practical operational aspects.

## References

1. EUROSTAT European Regional Statistics. Changes in the NUTS Classifications, 1981–1999. <http://ec.europa.eu/eurostat/documents/> (2002)
2. Council of European Municipalities and Regions Local and regional government in europe. Structure and Compatences. <http://www.ccre.org/> (2016)
3. Gazzetta Ufficiale Determinazione dei criteri per il riordino delle province. <http://www.gazzettaufficiale.it/eli/id/2012/11/06/012G0210/sg> (2012) (in Italian)
4. Gazzetta Ufficiale Disposizioni sulle città metropolitane, sulle province, sulle unioni e fusioni di comuni. Serie Generale. <http://www.gazzettaufficiale.it/eli/id/2014/4/7/14G00069/sg> (2014) (in Italian)
5. Drezner, Z., Hamacher, H.W. (eds.): Facility Location: Application and Theory. Springer, Berlin (2002)
6. Revelle, C.S., Eiselt, H.A.: Location analysis: a synthesis and survey. *Eur. J. Oper. Res.* **165**, 1–19 (2005)
7. Kalcsics, J., Nickel, S., Schröder, M.: Towards a unified territorial design approach—applications, algorithms and GIS integration. *Top*, **13**(1), 1–56. (2005)
8. Kalcsics, J.: Districting problems. In: Laporte, G., Nickel, S., Saldanha da Gama, F. (Eds.) *Location Science* (pp. 595–622). Springer International Publishing (2015)
9. ReVelle, C.S., Murray, A.T., Serra, D.: Location models for ceding market share and shrinking services. *Omega* **35**, 533–540 (2007)
10. Bruno, G., Esposito, E., Genovese, A., Piccolo, C.: Institutions and facility mergers in the Italian education system: Models and case studies. *Socio-Econ. Plann. Sci.* **53**, 23–32 (2016)
11. Bruno, G., Genovese, A., Piccolo, C.: Capacity management in public service facility networks: a model, computational tests and a case study. *Optim. Lett.* **10**(5), 975–995 (2016)
12. Bruno, G., Genovese, A., Piccolo, C., Sterle, C.: A location model for the reorganization of a school system: the italian case study. *Procedia-Social and Behavioral Sciences* **108**, 96–105 (2014)
13. Marsh, M.T., Schilling, D.A.: Equity measurement in facility location analysis: a review and framework. *Eur. J. Oper. Res.* **74**(1), 1–17 (1994)
14. Barbati, M., Piccolo, C.: Equality measures properties for location problems. *Optim. Lett.* **10**(5), 903–920 (2016)

15. Sterle, C., Sforza, A., Amideo, A.E.: Multi-period location of flow intercepting portable facilities of an intelligent transportation system. *Socio-Econ. Plann. Sci.* **53**, 4–13 (2016)
16. Sonmez, A.D., Lim, G.J.: A decomposition approach for facility location and relocation problem with uncertain number of future facilities. *Eur. J. Oper. Res.* **218**(2), 327–338 (2012)
17. Silva de Assis, L., Morelato Franca, P., Luiz Usberti, F.: A redistricting problem applied to meter reading in power distribution networks. *Comput. Oper. Res.* **41**(1), 65–75 (2013)
18. Bruno, G., Genovese, A., Piccolo, C.: Territorial amalgamation decisions in local government: models and a case study from Italy. *Socio-Econ. Plann. Sci.* **57**, 61–72 (2017)

# A Stochastic Maximal Covering Formulation for a Bike Sharing System

Claudio Ciancio, Giuseppina Ambrogio and Demetrio Laganá

**Abstract** This paper discusses a maximal covering approach for bike sharing systems under deterministic and stochastic demand. Bike sharing is constantly becoming a more popular and sustainable alternative transportation system. One of the most important elements for the design of a successful bike sharing system is given by the location of stations and bikes. The demands in each zone for each period is however uncertain and can only be estimated. Therefore, it is necessary to address this problem by taking into account the stochastic features of the problem. The proposed model determines the optimal location of bike stations, and the number of bikes located initially in each station, considering an initial investment lower than a given predetermined budget. The objective of the model is to maximize the percentage of covered demand. Moreover, during the time horizon, it is possible to relocate a certain amount of bikes in different stations with a cost proportional to the traveled distance. Both deterministic and stochastic models are formulated as mixed integer linear programs.

**Keywords** Bike sharing system · Stochastic model

## 1 Introduction

Due to the constant growth of interest over climate change a great attention is being given in the last years to possible alternative sustainable transportation modes. Bike sharing systems are an emerging mode of transportation that provide the temporary rental of bikes and have the potential to reduce car usage hence congestion [1].

---

C. Ciancio (✉) · G. Ambrogio · D. Laganá  
DIMEG - University of Calabria, Rende, Italy  
e-mail: claudio.ciancio@unical.it

G. Ambrogio  
e-mail: giuseppina.ambrogio@unical.it

D. Laganá  
e-mail: demetrio.lagana@unical.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_26

Moreover they have an impact on the promotion of an healthy lifestyle and on the reduction of pollution in the city. One of the most important factor for the success of a bike sharing system is the ability to react according to the variations in user demand [2]. Even if the daily demand on each location is characterized by random variations, there are relationships that can be identified and estimated through appropriate statistical analysis. The bike sharing system has to be planned and managed to maximize the level of customer satisfaction. Therefore, it is necessary to reduce as possible situations in which a user arrives at a station to take a bike and finds the station empty, or in which he arrives at the station to return the bike and the station is completely full. To this aim, it is necessary to analyse and solve opportunely strategic, tactical and operational problems. First, at a strategic level, it is necessary to define the location of the bike sharing stations. Second, at a tactical level, it is necessary to determine the number of docks and bikes located in each station. Last, at an operational level, a bike repositioning plan has to be defined for moving bikes, from one station with an excess to a station with a shortage, in order to satisfy the demand forecast for the next periods of the time horizon [3]. Different mathematical formulations can be used for the definition of the optimal location of the bike stations. Lin and Yang [4] proposed an integer non-linear problem to determine the optimal location of bike stations, based on cost minimization assuming a penalty for uncovered demand. However the model does not consider the relocation of bicycles but assumes that the number of bikes is always sufficient to satisfy the required demand. O'Mahony and Shmoys [5] addressed the problem of maintaining system balance during peak rush-hour usage as well as rebalancing overnight to prepare the system for rush-hour usage. Yan et al. [6] developed different models under deterministic and stochastic demands based on a time-space network to determine the locations of bike rental stations, bike fleet allocation and bike routing in which the objective function minimizes the sum of the operating costs of the bike rental system plus the fixed costs of locating the bike rental stations. Brinkmann et al. [7] proposed a multi-periodic inventory routing problem in which they took into account both time-dependent requests and target fill levels given by optimization models on the tactical management level. Kloimüllner et al. [8] introduced a problem where bike stations are marked for pick-ups or deliveries. Service times are considered as static and pick-up and delivery stations have to be visited alternately. A cluster-first route-second heuristic is implemented to assign stations to vehicles. Martinez et al. [9] formulated a mixed-integer linear model solved through a heuristic approach that allow optimizing the location of shared bike stations, assuming a fleet size and bicycle relocation calculation for a regular operating day. According to the model proposed by Frade and Ribeiro [10] we formulated the problem with a maximal covering location approach. The main decisions taken into account in the model here proposed are: the location of the public bike rental stations, the bike inventory levels of each station in each period and how the available not rented bikes have to be relocated during the investigated planning horizon. Therefore, even if the model is mainly developed as a support decision tool for strategic decisions (location

of stations, fleet size) it is also used to support operational decisions regarding bikes relocation. Both problems are however strictly related since the optimal relocation strategy is affected by the fleet size and the stations location.

## 2 Mathematical Formulation

The main goal of the mathematical model presented in this section is to maximize the percentage demand covered while taking into account constraints on the initial budget and required level of service. The model requires some input data such as the estimated demand for every period for each considered point of the graph, the maximum and minimum capacity of the stations, the price to locate stations and bicycles, the relocations costs and the available budget. The following notation has been used to represent the decision variables and input parameters, in the deterministic variant of the model:

### Sets

- $I$  set of demand zones, indexed by  $i$  and  $j$   
 $T$  set of time periods, indexed by  $t$

### Decision Variables

- $x_{ij}^t$  percentage of demand in the zone  $i$  in the period  $t$  covered by a station located in the zone  $j$   
 $y_i$  binary variable with value 1 if a bike station is activated in the zone  $i$ , 0 otherwise  
 $z_i$  number of blocks situated in the station in the zone  $i$   
 $v_i^t$  number of bikes located in the zone  $i$  at the period  $t$   
 $r_{ij}^t$  number of bikes relocated from  $i$  to  $j$  at the period  $t$

### Parameters

- $q_i^t$  demand in the zone  $i$  in the period  $t$   
 $z_{min}$  minimum number of bikes located in each block  
 $z_{max}$  maximum number of bikes located in each block  
 $c_s$  fixed cost of a station  
 $c_b$  unit price of a bicycle  
 $c_B$  unit price of a block  
 $c_r$  variable unit relocation trip cost  
 $d_{ij}$  distance from zone  $i$  to zone  $j$   
 $B$  initial budget  
 $k$  number of periods in the project horizon  
 $d_{max}$  maximum distance from the station

This notation has been used to define the following mathematical model

$$\max z = \sum_{i \in I} \sum_{j \in I} \sum_{t \in T} x_{ij}^t q_i^t \quad (1)$$

$$v_i^t = v_i^{t-1} + \sum_{j \in I} r_{ji}^{t-1} - \sum_{j \in I} r_{ij}^{t-1} \quad \forall i \in I \quad \forall t \in T \quad (2)$$

$$v_i^1 = v_i^k \quad \forall i \in I \quad (3)$$

$$x_{ij}^t = 0 \quad \forall (i,j) : d_{ij} > d_{max} \quad \forall t \in T \quad (4)$$

$$v_i^t \geq \sum_{j \in I} q_j^t x_{ji}^t \quad \forall i \in I \quad \forall t \in T \quad (5)$$

$$v_i^t \geq z_{min} z_i \quad \forall i \in I \quad \forall t \in T \quad (6)$$

$$v_i^t \leq z_{max} z_i \quad \forall i \in I \quad \forall t \in T \quad (7)$$

$$\sum_{j \in I} r_{ij}^t \leq v_i^t \quad \forall i \in I \quad \forall t \in T \quad (8)$$

$$\sum_{i \in I} (c_s y_i + c_B z_i + c_b v_i^1) + \sum_{i \in I} \sum_{j \in I} \sum_{t \in T} c_r r_{ij}^t d_{ij} \leq B \quad (9)$$

$$x_{ij}^t \leq y_j \quad \forall i \in I \quad \forall j \in I \quad \forall t \in T \quad (10)$$

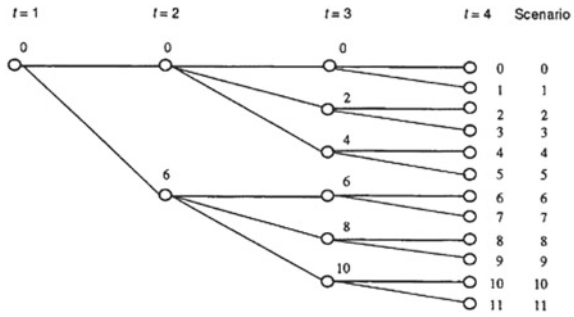
$$0 \leq x_{ij}^t \leq 1 \quad \forall i \in I \quad \forall j \in I \quad \forall t \in T \quad (11)$$

$$y_i, z_i \in \{0, 1\} \quad \forall i \in I \quad (12)$$

$$v_i^t, r_{ij}^t \in \mathbb{N}^+ \quad \forall i \in I \quad \forall j \in I \quad \forall t \in T \quad (13)$$

The objective function (1) maximize the total demand covered by the bike sharing system during the analyzed planning horizon. Constraints (2) are flow conservation constraints. In particular the number of bikes located in the zone  $i$  at time  $t$  is given by the number of bikes available in the previous time step and the total net flow of bikes relocated from or to  $i$  in the previous period. Equation (3) impose that the number of bikes at the beginning and at the end of the planning horizon is the same for each zone. This constraint is added since the model considers a periodic time horizon of one week. The demand of service in the zone  $i$  can be satisfied only by stations located in zones  $j$  at a distance  $d_{ij}$  less than a predetermined threshold  $d_{max}$  (4). The number of bikes located in the zone  $i$  at the period  $t$  has to be enough to satisfy the demand (5) and in a range defined according to the number of blocks located in the

Fig. 1 Scenario tree



station (6–7). The number of bikes relocated from the zone  $i$  has to be less or equal of the inventory level in the zone  $i$  at the same period  $t$  (8). The bike sharing system have to be compatible with the available budget  $B$  (9). Constraints (10) define the relationship between decision variables  $x$  and  $y$ . Finally Eqs. (11, 12, 13) define the decision variables domain.

### 2.1 Stochastic Model

In real contexts, operational decisions regard use of bikes are made for a number of decision points  $t \in T$ . In every decision point  $t$ , a solution have to be planned according to different demand scenarios. Every decision leads to a deterministic post-decision state. A stochastic transition then leads to different decision states for the next time period  $t + 1$ .

The stochastic model was formulated based on a scenario tree. For this version of the model the following decision variables were used Fig. 1:

- $x_{i,j}^n$  percentage of demand in the zone  $i$  in the period  $t(n)$  on the node  $n$  covered by a station located in the zone  $j$
- $y_i$  binary variable with value 1 if a bike station is activated in the zone in  $i$ , 0 otherwise
- $z_i$  number of blocks situated in the station in the zone  $i$
- $v_i^n$  number of bikes located in the zone  $i$  at the period  $t(n)$  on the node  $n$
- $r_{ij}^n$  number of bikes relocated from  $i$  to  $j$  at the time  $t(n)$  on the node  $n$

All the input parameters related to the time  $t$  will now refer to the correspondent node  $n$  of the scenario tree. For each node is defined a time  $t_n$ , a predecessor node  $\alpha_n$  and the number of child nodes  $\beta_n$ . We refer as  $p^n$  the probability with which node  $n$  occurs which is equal to  $p^n = \frac{p^{\alpha_n}}{\beta_{\alpha_n}}$ . Finally we indicate as  $N$  the node set and as  $N_l$  the set of leaf nodes. The stochastic model is formulated as follow:

$$\max z = \sum_{n \in N} \sum_{i \in I} \sum_{j \in I} x_{ij}^n q_i^n p^n \tag{14}$$



$$v_i^n = v_i^{\alpha_n} + \sum_{j \in I} r_{ji}^{\alpha_n} - \sum_{j \in I} r_{ij}^{\alpha_n} \quad \forall i \in I \quad \forall n \in N \quad (15)$$

$$v_i^1 = v_i^n \quad \forall i \in I \quad \forall n \in N_l \quad (16)$$

$$x_{ij}^n = 0 \quad \forall (i, j) : d_{ij} > d_{max} \quad \forall n \in N \quad (17)$$

$$v_i^n \geq \sum_{j \in I} q_j^n x_{ji}^n \quad \forall i \in I \quad \forall n \in N \quad (18)$$

$$v_i^n \geq z_{min} z_i \quad \forall i \in I \quad \forall n \in N \quad (19)$$

$$v_i^n \leq z_{max} z_i \quad \forall i \in I \quad \forall n \in N \quad (20)$$

$$\sum_{j \in I} r_{ij}^t \leq v_i^n \quad \forall i \in I \quad \forall n \in N \quad (21)$$

$$\sum_{i \in I} (c_s y_i + c_B z_i + c_b v_i^1) + \sum_{i \in I} \sum_{j \in I} \sum_{n \in N} p^n c_r r_{ij}^n d_{ij} \leq B \quad (22)$$

$$x_{ij}^n \leq y_j \quad \forall i \in I \quad \forall j \in I \quad \forall n \in N \quad (23)$$

$$0 \leq x_{ij}^n \leq 1 \quad \forall i \in I \quad \forall j \in I \quad \forall n \in N \quad (24)$$

$$y_i, z_i \in \{0, 1\} \quad \forall i \in I \quad (25)$$

$$v_i^n, r_{ij}^n \in \mathbb{N}^+ \quad \forall i \in I \quad \forall j \in I \quad \forall n \in N \quad (26)$$

### 3 Case Study

In this section, we discuss some computational experiments carried out to evaluate the performance of the proposed approach. Both deterministic and stochastic algorithms were implemented as single thread code in Java and solved through Cplex 12.6. All tests are performed on a desktop computer equipped with an Intel Core i7 processor with 2.4 GHz, 8 GB RAM, and running Windows 10. The mathematical model presented in the previous section was applied to the city of Cosenza in Italy. For this application we assumed the following assumptions: the maximum bike capacity of each station is proportional to the number of blocks (each block contains 10 docks) and is assumed to be 8 for each block while the minimum value is assumed to be 3. The instance is characterized by the following data reported in Table 1.

We considered an instance with 12 time periods. Each period represents a time length of 3 h. Bike relocation is allowed only at night (every 4 periods). Moreover we

**Table 1** Input data

Parameter	Value	Parameter	Value
Number of points	50	Bike station cost	3000€
Number of daily requests	1480	Bike cost	300€
Time horizon	12 periods	Bike docks	400€
Number of scenarios	2048		

impose that the bike inventory is the same for each station at the end of the first day (after 4 periods) and at the end of the time horizon. This assumption is performed since we consider the first 4 periods of time to represent a weekday and the other 8 periods for Saturday and Sunday.

### 3.1 Demand Analysis

One of the main problems for the BSS design is the estimation of the potential demand to the service. Survey questions were designed to determine the nature of the trip and possible destinations. Specifically, the aim of the survey was to determine: what travels (starting and end location) and how many users would be attracted by a BSS; the maximum distance that a user would be willing to travel to reach the closest station; the frequency and time with which each user will make use of the system; the average time of each trip. In addition, we asked demographic questions related to sex, income, and age.

### 3.2 Numerical Results

The stochastic problems has been solved considering the data collected from the survey and subsequently through the use of a scenario generation technique. We considered three different configurations related to different initial budgets between 300000€ and 700000€. The results of the optimal solution given by the model are reported in Table. 2.

For the first test (initial budget of €300000) the system covers 748 daily requests (50.5% of the total demand) defining the location of 16 stations in the geographic area and it makes use of a total amount of 192 bikes. The total investment is €298500 (slightly less than the available budget). In the second test, we assumed a budget of €500000. The optimal solution, covers 1128 daily requests (76.2% of the total

**Table 2** Numerical result of the stochastic model with different budgets

Initial investment	€300000	€500000	€700000
Number of trips covered	748	1128	1480
Number of stations located	16	28	41
Total number of docks	240	450	665
Total number of bikes	192	360	532
Total cost	€298500	€498200	€699600
VSS	33	51	28

demand) and locates 28 stations. Finally, in the last test, the initial budget is increased to €700000. The optimal solution covers all 1480 daily requests, locates 41 stations and the fleet contains 532 bikes. To measure the quality of the stochastic model compared to the deterministic version we used the VSS indicator (Value of the Stochastic Solution). The value we obtained ranged between 28 and 51 additional covered requests. It is interesting to note that in the last test the system is able to cover all 1480 requests using the stochastic version of the model while 28 requests are not satisfied by solving the correspondent deterministic problem.

## 4 Conclusions

This paper discusses a mathematical formulation to determine the optimal location of a set of bike-sharing stations, the number of bikes allocated in each station and can be used to estimate a short-time operational plan and the related cost based on the demand estimated through surveys. The problem has been addressed by means of a maximum coverage model subjected to constraints related to the initial budget and on a minimum level of service measured based on the distance between user and station. The proposed methodology can provide a decision support tool to urban managers and therefore it has the potential to contribute significantly to the quality of the bike-sharing system. Moreover, the stochastic formulation allows identifying the optimal bike relocation plan according to different possible demand scenarios during an investigated time horizon. The approach was used on a case study for the city of Cosenza in Italy. The model was solved considering different operational constraints and initial budget levels. The results show that the stochastic model gives better indications compared to the equivalent deterministic variant.

## References

1. Frade, I., Ribeiro, A.: Bicycle sharing systems demand. *Procedia-Soc. Behav. Sci.* **111**, 518–527 (2014)
2. Alvarez-Valdes, R., Belenguer, J.M., Benavent, E., Bermudez, J.D., Muñoz, F., Vercher, E., Verdejo, F.: Optimizing the level of service quality of a bike-sharing system. *Omega* **62**, 163–175 (2016)
3. Shu, J., Chou, M.C., Liu, Q., Teo, C.P., Wang, I.L.: Models for effective deployment and redistribution of bicycles within public bicycle-sharing systems. *Oper. Res.* **61**(6), 1346–1359 (2013)
4. Lin, J.R., Yang, T.H.: Strategic design of public bicycle sharing systems with service level constraints. *Transp. Res. Part E: Logist. Transp. Rev.* **47**(2), 284–294 (2011)
5. O'Mahony, E., Shmoys, D.B.: Data analysis and optimization for (Citi) bike sharing. In: *AAAI*, pp. 687–694. (2015)
6. Yan, S., Lin, J.R., Chen, Y.C., Xie, F.R.: Rental bike location and allocation under stochastic demands. *Comput. Ind. Eng.* **107**, 1–11 (2017)
7. Brinkmann, J., Ulmer, M.W., Mattfeld, D.C.: Inventory routing for bike sharing systems. *Transp. Res. Procedia* **19**, 316–327 (2016)
8. Kloimüller, C., Papazek, P., Hu, B., Raidl, G.R.: A cluster-first route-second approach for balancing bicycle sharing systems. In: *Computer Aided Systems Theory EUROCAST Lecture Notes in Computer Science*, vol. 95(20), pp. 439–446. Springer (2015)
9. Martinez, L.M., Caetano, L., Eiró, T., Cruz, F.: An optimisation algorithm to establish the location of stations of a mixed fleet biking system: an application to the city of Lisbon. *Procedia Soc. Behav. Sci.* **54**, 513–524 (2012)
10. Frade, I., Ribeiro, A.: Bike-sharing stations: a maximal covering location approach. *Transp. Res. Part A: Policy Pract.* **82**, 216–227 (2015)

# A Model and Algorithm for Solving the Landfill Siting Problem in Large Areas

Mariano Gallo

**Abstract** In this paper we study the problem of siting landfills over large geographical areas. An optimisation problem is formulated and solved with a heuristic algorithm. The formulation of the problem explicitly considers economic compensation for the population of areas affected by the landfill. The approach is used to solve the problem in the real-scale case of the southern Italian region of Campania, with 551 possible sites. The proposed methodology is able to solve the problem with acceptable computing times.

**Keywords** Landfills · Location problems · Waste management

## 1 Introduction

Finding suitable locations for landfills, incinerators or other major waste management sites constitutes a major and somewhat thorny problem for public decision makers, especially at regional level. Besides technical and economic considerations, a critical issue concerns the strong opposition of local communities that live in the area chosen for siting the facility: the well-known NIMBY (Not In My Back Yard) syndrome. This phenomenon in some cases can spark violent demonstrations and road blockades. Since a waste management facility, especially a landfill, can actually lead to a reduction in property values in adjacent zones, economic compensation should be paid to residents in the areas affected by the facility.

Several approaches to the problem of landfill site location have been proposed in the literature. Waste management models were reviewed by Morrissey and Browne [1], while multi-criteria approaches were proposed by Melachrinoudis et al. [2], Hokkanen and Salminen [3], Cheng et al. [4], Vasiloglou [5], Kontos et al. [6], Chang et al. [7], Xi et al. [8], Gorsevski et al. [9] and Gbanie et al. [10]. Al-Jarrah

---

M. Gallo (✉)

Dipartimento di Ingegneria, Università del Sannio, Piazza Roma 21,  
82100 Benevento, Italy  
e-mail: gallo@unisannio.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_27

267

and Abu-Qdais [11] proposed a system based on fuzzy inference, while Bennis and Bahi [12] studied criteria for landfill location. GIS-based procedures were proposed by Sumathi et al. [13], Zamorano et al. [14] and Simsek et al. [15]. A comparison between actual locations and optimal locations of landfills was attempted by Eiselt [16], while the joint location of landfills and transfer stations was proposed by Eiselt and Marianov [17]. By contrast, Guiqin et al. [18] proposed an analytical hierarchy process (AHP) for landfill site selection.

In this paper we formulate an optimisation model for solving the landfill location problem, taking explicitly into account: (a) transportation costs from waste production sites to landfills; (b) construction and maintenance costs of the facility; (c) the costs of economic compensation for the residents in adjacent areas. While costs (a) and (b) are usually considered in almost all waste facility siting models, costs (c) are usually not explicitly considered.

This problem can be seen as a  $p$ -median problem [19]; Daskin and Maass, [20]; a literature review of  $p$ -median problems can be found in ReVelle et al. [21]. For solving the model, we propose a heuristic algorithm that is able to give a good, albeit sub-optimal, solution in an acceptable computing time also for large real-scale problems. This algorithm is a variant of a greedy heuristic [22] coupled with a multi-start technique [23]. This paper focuses mainly on the proposed model for solving or limiting the Nimby problem, and on the case study; the model can be solved with other algorithms proposed in the literature: a comparison among algorithms is outside the scope of the paper.

Below we make reference to landfills, since they are the facilities that generate the strongest opposition from local communities and for which economic compensation should be highest. However, the model can be easily extended to the siting of other “undesirable” plants. Moreover, we examine a region where waste sources are represented by the municipalities and the subjects affected by economic compensation are all people (households) living in each municipality (except for large cities, which can be partitioned into zones). The model can be easily extended to smaller areas for different kinds of plants.

## 2 Problem Description and Model Formulation

We assume that several landfills have to be located in a geographically large area, such as a region, to satisfy waste disposal requirements, and that the location of all waste sources is known, as is the annual waste production of each source. Moreover, all eligible sites in the area are initially identified and the cost of each site depends on the quantity of waste that it has to treat each year (construction costs are appropriately amortised and attributed on an annual basis). In addition, the compensation costs for the residents concerned are then considered. Finally, the cost per ton-km of transported refuse is known. The constraints of the problem concern the kind of variables and the maximum number of facilities (the minimum is, obviously, equal to 1). The objective function, to be minimised, is the total yearly cost, comprising the

sum of transportation costs, construction/maintenance costs and compensation costs. Finally, this problem is NP-hard like almost all facility siting problems [24].

The decision variables of the problem are binary,  $x_i = 0/1$ , where  $i$  indicates an eligible site and  $x_i = 1$  if a landfill is envisaged in site  $i$ ,  $x_i = 0$  otherwise; they may be organised in a vector,  $\mathbf{x}$ . We indicate with  $m$  the maximum number of landfills envisaged for the area and with  $n$  the number of eligible sites. We assume that, for each possible solution, the waste produced by a town is allocated to the nearest landfill, that is:

$$w_i(\mathbf{x}) = \sum_j wp_j \cdot a_{i,j}(\mathbf{x})$$

where  $wp_j$  is the annual waste production of town  $j$  and  $a_{i,j}$  is equal to 1 if the distance between  $i$  and  $j$ ,  $d_{i,j}$ , on the road network is the minimum compared to all distances between  $j$  and any other site  $i$ . Below,  $d_{j,min}$  is used to indicate this minimum distance and hence  $a_{i,j} = 1$  if  $d_{i,j} = d_{j,min}$ .

The annual cost,  $yc_i$ , of each landfill is known and is represented by a function,  $yc_i(w_i(\mathbf{x}))$ , that has to be calibrated. We use  $in_i$  to indicate the number of inhabitants in town  $i$  (or in zone  $i$  for cities) and  $cc$  for the annual compensation cost per ton of waste per inhabitant. Finally,  $ctkm$  is the cost per ton-km of transported waste, that we assume independent of the kind of roads to be travelled along (the problem can be easily extended to roads with different running costs, acting on the calculation of  $d_{j,min}$ ).

The objective function is the sum of three terms: the first is the annual construction/maintenance cost of the landfills, the second is the cost of transportation and the last is the compensation cost. Under these assumptions, the optimisation model can be formulated as follows:

$$\mathbf{x}^{opt} = \text{Arg}_{\mathbf{x}} \min \left( \sum_i yc_i(w_i(\mathbf{x}) \cdot x_i) + \sum_j d_{j,min}(\mathbf{x}) \cdot wp_j \cdot ctkm + \sum_i in_i \cdot w_i(\mathbf{x}) \cdot cc \right)$$

$$\text{s.t.} \quad x_i = 0/1 \quad \forall i$$

$$1 \leq \sum_i x_i \leq m$$

This model is a binary non-linear constrained optimisation model: function  $yc_i(w_i(\mathbf{x}))$  may not be linear and the values  $d_{j,min}(\mathbf{x})$  change with  $\mathbf{x}$  in a non-linear manner since, with each solution, the destination  $i$  for the waste produced by town  $j$  may change. Analogously, the terms  $w_i(\mathbf{x})$  change non-linearly with  $\mathbf{x}$ .

### 3 Solution Algorithm

In order to solve this problem we propose a heuristic algorithm based on exhaustive mono-dimensional searches and a multi-start procedure. First of all, the proposed algorithm is simpler to manage if we organise the decision variables differently as

**Fig. 1** An example of variables transformation

$$\begin{array}{c}
 \boxed{
 \begin{array}{c}
 x = [0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0] \\
 \Downarrow \\
 \chi = [3, 7, 8, 12, 0]
 \end{array}
 }
 \end{array}$$

follows: (a) an integer number is associated to each landfill site  $i$ ; (b)  $m$  pointer variables,  $\chi_k$  (collected in vector  $\chi$ ), are defined; each pointer variable can assume an integer value between 0 and  $n$  which indicates the corresponding site  $i$  or, if  $\chi_k = 0$ , that this pointer variable is not associated to any site  $i$ . In this way, each solution of the problem can be represented by the values assumed by the pointer variables. In Fig. 1 an example with  $n = 20$  and  $m = 5$  is reported.

There are two advantages of this transformation: (1) the constraint on the maximum number of landfills is automatically included in the problem formulation (and also that on the minimum number, imposing that with 0 landfills the transportation cost is infinite); (2) the space of all solutions is limited only to those that respect this constraint, since solutions with more than  $m$   $x_i = 1$  are *ipso facto* excluded.

The proposed algorithm works on the pointer variables to identify a solution to the optimisation problem. With the variable transformation the optimisation problem is modified as follows:

$$\begin{aligned}
 \chi^{opt} = \text{Arg}_{\chi} \min & \left( \sum_i y c_i (w_i(\chi) \cdot x_i(\chi)) + \sum_j d_{j,min}(\chi) \cdot w p_j \cdot c t k m + \sum_i i n_i \cdot w_i(\chi) \cdot c c \right) \\
 \text{s.t.} \quad & 0 \leq \chi_k \leq n \quad \forall k \\
 & \chi_k \text{ integer} \quad \forall k \\
 & \sum_k \chi_k \geq 1
 \end{aligned}$$

The proposed algorithm is articulated in the following phases:

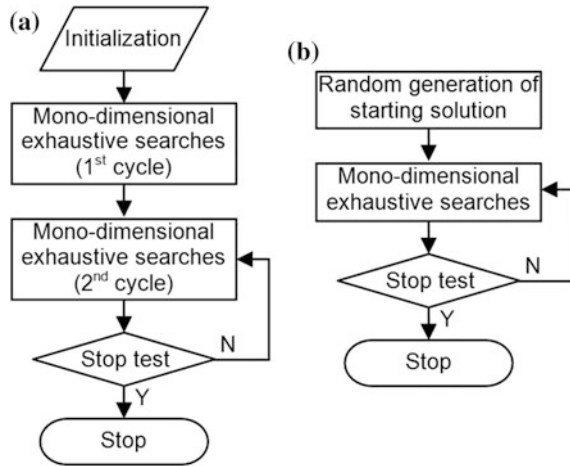
*Phase 0—Initialisation.* All variables  $\chi_k$  are set equal to 0, the counter of iterations is fixed to 0 and the value of the objective function is assumed infinite.

*Phase 1—Mono-dimensional exhaustive search (1st cycle).* A mono-dimensional exhaustive search is performed, examining all solutions obtained by changing only the first pointer value from 0 to  $n$ ; having identified the best value, the first pointer is fixed and the mono-dimensional exhaustive search is performed for the second pointer and so on. This phase ends either when the optimal value of a pointer is equal to 0 (no other landfills) or when all pointers have been exhaustively explored. At the end of this phase, the algorithm tested at most  $m \cdot n$  solutions.

*Phase 2—Mono-dimensional exhaustive search (2nd cycle).* Mono-dimensional exhaustive searches are performed as in the 1st cycle, starting from the values obtained at the end of the previous phase. If the best objective function value at the end of this phase is equal to the best value obtained in the previous phase, the search ends; otherwise, phase 2 is repeated until the stop condition is reached or a maximum number of solutions have been examined.



Fig. 2 Algorithm framework



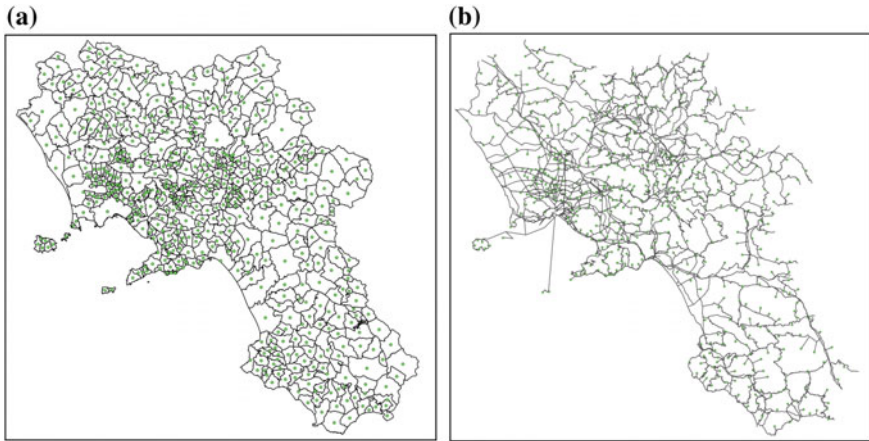
The general scheme of the algorithm is reported in Fig. 2a. This algorithm is able to generate a local optimum: all the neighbouring solutions, obtained by changing the value of any pointer, are worse than the solution obtained with the algorithm. Since the problem is not convex, in order to explore the solution space, we propose to test the same algorithm starting from randomly generated different initial solutions, instead of all pointers equal to 0. In such cases, only the second cycle has to be performed (see Fig. 2b). The complexity of each mono-dimensional exhaustive search is  $O(m \cdot n)$ .

### 4 Numerical Results

The proposed model and algorithm were tested on a real-scale case, namely the region of Campania, to ascertain the applicability of the proposed approach to large-scale problems. In this test we used the following data:

- total waste produced by each municipality in the region was obtained from official ISPRA data [25];
- average transportation cost was assumed equal to 0.4 €/t-km, obtained from some regional tenders for commissioning waste transport;
- the annual compensation cost was assumed equal to 0.0001 €/t per inhabitant;
- all 551 municipalities of Campania were considered candidate sites for the landfills and distances were measured between the centroids representing the municipalities (see Fig. 3a);
- the maximum number of landfills is equal to 5.

The distance matrix, required for estimating the transportation costs, was generated by implementing with the software Omnitrans 6 the regional road network;



**Fig. 3** Case study: (a) municipalities and centroids; (b) road network

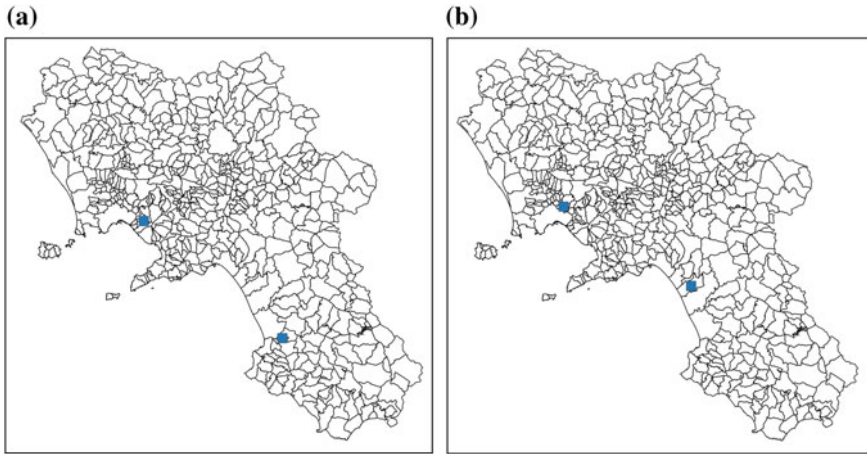
the network graph comprises over 20,000 mono-directional links representing over 6,000 km of roads (Fig. 3b). The annual cost of a landfill was calculated with the following formula Eiselt and Marianov [17]:

$$yc_i(w_i(\chi)) = c_0 + w_i(\chi) \cdot c_1(w_i(\chi))$$

where  $c_0$  is a constant cost (€/year) of the landfill, independent of  $w_i$ , and  $c_1(\cdot)$  is a variable cost (€/t-year) that depends on  $w_i$  for considering the scale economy. The parameters of this function should be calibrated with real data; in these initial results we used the parameters proposed by Eiselt and Marianov [17], appropriately converted to euros:  $c_0 = \text{€}220,000$  and  $c_1(w_i(\chi)) = 428.015 \cdot w_i(\chi)^{-0.209}$ .

The proposed algorithm, starting from all pointers equal to 0, examined 1654 solutions in about 21 min. The suboptimal solution thus obtained identifies two landfills, located in the municipalities of Massa di Somma and Giungano. The value of the objective function is €89,898,825 per annum, made up of transportation costs (€34,274,833/year), construction/maintenance costs (€54,311,274/year) and compensation costs (€1,312,718/year). The multi-start procedure, starting from 15 random solutions, led to the same local optimal solution. We tested the same procedure without considering the compensation costs; in this case two landfills were identified in the municipalities of Casoria and Battipaglia and the value of the objective function is €86,024,933 per annum, made up of transportation costs (€30,270,389/year) and of construction/maintenance costs (€55,754,544/year). In Fig. 4 the locations are reported for both cases.

In order to evaluate the quality of the solution, an exhaustive search was performed among all solutions that have only two landfills (151,525 solutions, computing time about 32 h); this search led to the same result obtained with the proposed algorithm.



**Fig. 4** Optimal landfill location: (a) with comp. costs; (b) without comp. costs

## 5 Conclusions

In this paper a model and an algorithm were proposed for solving the landfill siting problem. The procedure was tested on a real-scale case, the southern Italian region of Campania with 551 possible site locations. The problem is formulated by considering also compensation costs for the resident population affected by the facility in order to avoid or limit the NIMBY syndrome. Initial results highlight the applicability of the proposed approach and the different solutions obtainable with and without considering the compensation costs in the objective function.

Future research will seek to improve the model by considering actual possible locations and actual construction/maintenance costs, and proposing methods for defining compensation costs, considering the distances of inhabitants from landfills, albeit residents in other municipalities. A comparison with other algorithms proposed in the literature, in terms of computing times and goodness of the solution, and with some optimisation software will be object of future research.

## References

1. Morrissey, A.J., Browne, J.: Waste management models and their application to sustainable waste management. *Waste Manag.* **24**, 297–308 (2004)
2. Melachrinoudis, E., Min, H., Wu, X.: A multiobjective model for the dynamic location of landfills. *Location Sci.* **3**, 143–166 (1995)
3. Hokkanen, J., Salminen, P.: Choosing a solid waste management system using multicriteria decision analysis. *Eur. J. Oper. Res.* **98**, 19–36 (1997)

4. Cheng, S., Chan, C.W., Huang, G.H.: An integrated multi-criteria decision analysis and inexact mixed integer linear programming approach for solid waste management. *Eng. Appl. Artif. Intell.* **16**, 543–554 (2003)
5. Vasiloglou, V.C.: New tool for landfill location. *Waste Manag. Res.* **22**, 427–439 (2004)
6. Kontos, T.D., Komilis, D.P., Halvadakis, C.P.: Siting MSW landfills with a spatial multiple criteria analysis methodology. *Waste Manag.* **25**, 818–832 (2005)
7. Chang, N.-B., Parvathinathan, G., Breeden, J.B.: Combining GIS with fuzzy multicriteria decision-making for landfill siting in a fast-growing urban region. *J. Environ. Manage.* **87**, 139–153 (2008)
8. Xi, B.D., Su, J., Huang, G.H., Qin, X.S., Jiang, Y.H., Huo, S.L., Ji, D.F., Yao, B.: An integrated optimization approach and multi-criteria decision analysis for supporting the waste-management system of the City of Beijing, China. *Eng. Appl. Artif. Intell.* **23**, 620–631 (2010)
9. Gorsevski, P.V., Donevska, K.R., Mitrovski, C.D., Frizado, J.P.: Integrating multi-criteria evaluation techniques with geographic information systems for landfill site selection: a case study using ordered weighted average. *Waste Manag.* **32**, 287–296 (2012)
10. Gbanie, S.P., Tengbe, P.B., Momoh, J.S., Medo, J., Kabba, V.T.S.: Modelling landfill location using geographic information systems (GIS) and multi-criteria decision analysis (MCDA): case study Bo, Southern Sierra Leone. *Appl. Geogr.* **36**, 3–12 (2013)
11. Al-Jarrah, O., Abu-Qdais, H.: Municipal solid waste landfill siting using intelligent system. *Waste Manag.* **26**, 299–306 (2006)
12. Bennis, K., Bahi, L.: PCA and cluster analysis for criteria mapping in landfill siting. *Int. J. Eng. Technol.* **6**, 2244–2260 (2014)
13. Sumathi, V.R., Natesan, U., Sarkar, C.: GIS-based approach for optimized siting of municipal solid waste landfill. *Waste Manag.* **28**, 2146–2160 (2008)
14. Zamorano, M., Moleró, E., Hurtado, A., Grindlay, A., Ramos, A.: Evaluation of a municipal landfill site in Southern Spain with GIS-aided methodology. *J. Hazard. Mater.* **160**, 473–481 (2008)
15. Simsek, C., Elci, A., Gunduz, O., Taskin, N.: An improved landfill site screening procedure under NIMBY syndrome constraints. *Landscape Urban Plann.* **132**, 1–15 (2014)
16. Eiselt, H.A.: Locating landfills-optimization vs. reality. *Eur. J. Oper. Res.* **179**, 1040–1049 (2007)
17. Eiselt, H.A., Marianov, V.: A bi-objective model for the location of landfills for municipal solid waste. *Eur. J. Oper. Res.* **235**, 187–194 (2014)
18. Guiqin, W., Li, Q., Guoxue, L., Lijun, C.: Landfill site selection using spatial information technologies and AHP: A case study in Beijing, China. *J. Environ. Manage.* **90**, 2414–2421 (2009)
19. Kariv, O., Hakimi, L.: An algorithmic approach to network location problems, part ii: the p-Medians. *SIAM J. Appl. Math.* **37**, 539–560 (1979)
20. Daskin, M.S., Mass, K.L.: The p-Median problem. Laporte G., Nickel S., Saldanha da Gama F. (eds.), In: *Location Science*, pp. 21–45. Springer, Switzerland (2015)
21. ReVelle, C.S., Eiselt, H.A., Daskin, M.S.: A bibliography of some fundamental problem categories in discrete location science. *Eur. J. Oper. Res.* **184**, 817–848 (2008)
22. Whitaker, R.: A fast algorithm for the greedy interchange of large-scale clustering and median location problems. *INFOR* **21**, 95–108 (1983)
23. Resende, M.G.C., Werneck, R.F.: A hybrid heuristic for the p-Median problem. *J. Heuristics* **10**, 59–88 (2004)
24. Hochbaum, D.S.: When are NP-hard location problems easy? *Ann. Oper. Res.* **1**, 201–214 (1984)
25. ISPRA (2015) Catasto Rifiuti [web: <http://www.catasto-rifiuti.isprambiente.it/index.php?pg=provincia&aa=2015&regid=Campania>—visited 25/02/2017]

# Optimal Content Distribution and Multi-resource Allocation in Software Defined Virtual CDNs

Jaime Llorca, Antonia M. Tulino, Antonio Sforza and Claudio Sterle

**Abstract** A software defined virtual content delivery network (SDvCDN) is a virtual cache network deployed fully in software over a programmable cloud network infrastructure that can be elastically consumed and optimized using global information about network conditions and service requirements. We formulate the joint content-resource allocation problem for the design of SDvCDNs, as a minimum cost mixed-cast flow problem with resource activation decisions. Our solution optimizes the placement and routing of content objects along with the allocation of the required virtual storage, compute, and transport resources, capturing activation and operational costs, content popularity, unicast and multicast delivery, as well as capacity and latency constraints. Numerical experiments confirm the benefit of elastically optimizing the SDvCDNs configuration, compared to the dedicated provisioning of traditional CDNs.

**Keywords** Softwared defined virtual CDN · Flow-location-routing problem

---

J. Llorca · A.M. Tulino  
Nokia Bell Labs, Crawford Hill, NJ, USA  
e-mail: jaime.llerca@nokia-bell-labs.com

A.M. Tulino  
e-mail: a.tulino@nokia-bell-labs.com

A. Sforza · C. Sterle (✉) · A.M. Tulino  
Department of Electrical Engineering and Information Technology,  
University of Naples, Naples, Italy  
e-mail: claudio.sterle@unina.it

A. Sforza  
e-mail: sforza@unina.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_28

## 1 Introduction

With the soaring number and heterogeneity of video-capable devices, high quality video content, and video-based cloud services, communication networks are faced with increasingly tighter requirements (e.g., several Gbps bandwidth and a few ms latency). In this challenging environment, operators can leverage distributed cloud networking technologies that enable the deployment of a wide range of services in the form of software functions instantiated over general purpose hardware at multiple cloud locations distributed throughout the network [1, 2]. In this context, network functions virtualization (NFV) and software defined networking (SDN) play a key driving role. NFV enables porting network functions, today residing on dedicated hardware platforms, to a virtualized cloud infrastructure; while SDN allows programmatically configuring the network that interconnects the distributed cloud locations [1]. The confluence of NFV and SDN hence enables a highly adaptable cloud network platform that provides dynamic allocation of resources and choreographed inter-networking.

A software defined virtual content distribution network, or SDvCDN, is a virtual cache network deployed fully in software over a programmable cloud network infrastructure that can be elastically consumed and optimized using global information about network conditions and service requirements. In a SDvCDN, both virtual caches at distributed cloud locations as well as their virtual connectivity can be adaptively configured in order to meet service requirements with minimum overall use of the physical infrastructure. In this work, we focus on the end-to-end configuration of SDvCDNs, a problem that involves jointly optimizing the placement and routing of content objects, as well as the allocation of storage, compute, and transport resources. We develop a novel minimum cost mixed-cast flow formulation for the joint content-resource allocation problem in SDvCDNs that integrates placement, routing, and (virtual) resource allocation decisions in the form of a mixed integer linear program (MILP). These three problems have been separately considered in previous works:

1. *Data placement problem (DPP)*. It is an NP-hard problem introduced in [3] that consists in determining the placement of data objects in an arbitrary network with capacity constrained caches, such that user requests are satisfied with minimum total access cost. Each object request is assumed to be served by the closest replica without considering routing optimization nor transport capacity constraints. A number of approximation algorithms (e.g., [3, 4]) and greedy algorithms (e.g., [5–8]) have been proposed under several assumptions, such as uniform object sizes, network symmetry, and/or hierarchical topologies.
2. *Content distribution problem (CDP)*. It is a generalization of the DPP that integrates placement and routing decisions in an arbitrary capacitated cache network. The goal is to minimize both transport and storage costs while satisfying possible delivery deadline constraints [9].
3. *Virtual network embedding problem (VNE)*. It is an NP-Hard problem whose goal is to optimize the allocation of resources from a substrate network to vir-

tual network requests that specify a set of virtual nodes, virtual links, and traffic demands [10]. As opposed to the CDP, VNE addresses the resource allocation problem, but the placement of data sources is given as an input to the problem.

While in traditional CDNs, the allocation of dedicated physical appliances happens at a much longer time-scale than the actual placement and routing of content objects, the increased elasticity of SDvCDNs offers overall efficiency improvements by jointly optimizing content distribution and resource allocation [11–15]. In this paper, we formally address the joint content and multi-resource allocation problem in SDvCDNs. Our solution jointly optimizes the placement and routing of content objects along with the allocation of storage, compute, and transport resources, applies to arbitrary network topologies, and captures activation and operational costs, popularity settings, unicast and multicast delivery, as well as capacity and latency constraints. Our work extends the concepts and preliminary results presented by the authors in [16].

The remainder of this paper is organized as follows: the problem setting and MILP model are presented in Sect. 2; Sect. 3 describes numerical results on large-scale network settings; finally, conclusions are given in Sect. 4.

## 2 System Model and Problem Formulation

We model a distributed cloud network as a directed graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  with  $|\mathcal{V}|$  nodes representing distributed cloud locations in which virtual caches can be instantiated over general purpose servers, and  $|\mathcal{E}|$  edges representing the logical (e.g., IP) connectivity over which virtual links can be instantiated.

We formulate the joint content-resource allocation problem in SDvCDNs as a mixed-cast flow network design problem with the following input parameters and decision variables:

### *Input Parameters*

- $e_i^{st}$ ,  $e_i^{pr}$ , and  $e_{ij}^{tr}$ : efficiency (in terms of costs) of the storage, compute, and transport resources at node  $i \in \mathcal{V}$  and link  $(i, j) \in \mathcal{E}$ , respectively
- $c_{i,k}^{st}$  and  $a_{i,k}^{st}$ : capacity and activation cost of  $k \in \mathcal{H}_i^{st}$  storage servers at node  $i \in \mathcal{V}$ , where  $\mathcal{H}_i^{st}$  is the set of integers denoting the possible number of active storage servers at node  $i \in \mathcal{V}$
- $c_{i,k}^{pr}$  and  $a_{i,k}^{pr}$ : capacity and activation cost of  $k \in \mathcal{H}_i^{pr}$  compute servers at node  $i \in \mathcal{V}$ , where  $\mathcal{H}_i^{pr}$  is the set of integers denoting the possible number of active compute servers at node  $i \in \mathcal{V}$
- $c_{ij,k}^{tr}$  and  $a_{ij,k}^{tr}$ : capacity and activation cost of  $k \in \mathcal{H}_{ij}^{tr}$  IP links between nodes  $i$  and  $j$ , where  $\mathcal{H}_{ij}^{tr}$  is the set of integers denoting the possible number of active IP links between  $i$  and  $j$
- $\lambda_{d,o}$ : demand for content object  $o \in \mathcal{O}$  at destination  $d \in \mathcal{V}$
- $q_i^{d,o}$ : binary parameter that takes value 1 if  $\lambda_{d,o} > 0$  and  $i = d$ ; 0 otherwise

- $B_o$ : size of content object  $o \in \mathcal{O}$
- $H_{d,o}$ : maximum acceptable delay for the delivery of object  $o \in \mathcal{O}$  to node  $d \in \mathcal{V}$

### Decision Binary Variables

- User-object flows:  $f_i^o$  indicates the maximum amount of object  $o$  at node  $i \in \mathcal{V}$ ;  $f_i^{d,o}$  indicates the fraction of object  $o$  stored and processed at node  $i \in \mathcal{V}$  for destination  $d \in \mathcal{V}$ ;  $f_{ij}^{d,o}$  the fraction of object  $o \in \mathcal{O}$  carried by link  $(i,j)$  for destination  $d \in \mathcal{V}$ .
- Global flows:  $f_i^{st}$  is the total amount of information stored at node  $i \in \mathcal{V}$ ,  $f_i^{pr}$  is the total amount of information processed at node  $i \in \mathcal{V}$ ;  $f_{ij}^{tr}$  is the total amount of information carried by link  $(i,j)$ .
- Resource activation variables:  $y_{i,k}^{st} = 1$  indicates the activation of  $k$  storage servers at node  $i \in \mathcal{V}$ ,  $y_{i,k}^{pr} = 1$  the activation of  $k$  compute servers at node  $i \in \mathcal{V}$ ;  $y_{ij,k} = 1$  the activation of  $k$  IP links between nodes  $i$  and  $j$ .

The joint content-resource allocation problem is then formulated as follows:

minimize:

$$\sum_{i \in \mathcal{V}} \left( e_i^{st} f_i^{st} + \sum_{k \in \mathcal{K}_i} a_{i,k}^{st} y_{i,k}^{st} \right) + \sum_{i \in \mathcal{V}} \left( e_i^{pr} f_i^{pr} + \sum_{k \in \mathcal{K}_i} a_{i,k}^{pr} y_{i,k}^{pr} \right) + \sum_{(i,j) \in \mathcal{E}} \left( e_{ij}^{tr} f_{ij}^{tr} + \sum_{k \in \mathcal{K}_{ij}} a_{ij,k}^{tr} y_{ij,k}^{tr} \right) \quad (1)$$

s.t.

$$q_i^{d,o} + \sum_{j \in \delta^+(i)} f_{ij}^{d,o} = f_i^{d,o} + \sum_{j \in \delta^-(i)} f_{ji}^{d,o} \quad \forall i \in \mathcal{V}, d \in \mathcal{V}, o \in \mathcal{O} \quad (2)$$

$$f_i^{d,o} \leq f_i^o \quad \forall i \in \mathcal{V}, d \in \mathcal{V} \quad (3)$$

$$\sum_{o \in \mathcal{O}} f_i^o B_o = f_i^{st} \quad \forall i \in \mathcal{V} \quad (4)$$

$$\sum_{o \in \mathcal{O}} \sum_{d \in \mathcal{V}} f_{ij}^{d,o} \lambda_{d,o} B_o = f_{ij}^{tr} \quad \forall (i,j) \in \mathcal{E} \quad (5)$$

$$\sum_{o \in \mathcal{O}} \sum_{d \in \mathcal{V}} f_i^{d,o} \lambda_{d,o} B_o = f_i^{pr} \quad \forall i \in \mathcal{V} \quad (6)$$

$$f_i^{st} \leq \sum_{k \in \mathcal{K}_i} c_{i,k}^{st} y_{i,k}^{st} \quad \forall i \in \mathcal{V} \quad (7)$$

$$f_i^{pr} \leq \sum_{k \in \mathcal{K}_i} c_{i,k}^{pr} y_{i,k}^{pr} \quad \forall i \in \mathcal{V} \quad (8)$$

$$f_{ij}^{tr} \leq \sum_{k \in \mathcal{K}_{ij}} c_{ij,k}^{tr} y_{ij,k}^{tr} \quad \forall (i,j) \in \mathcal{E} \quad (9)$$



$$\sum_{k \in \mathcal{K}_i} y_{i,k}^{st} \leq 1 \quad \forall i \in \mathcal{V} \quad (10)$$

$$\sum_{k \in \mathcal{K}_i} y_{i,k}^{pr} \leq 1 \quad \forall i \in \mathcal{V} \quad (11)$$

$$\sum_{k \in \mathcal{K}_{ij}} y_{ij,k}^{tr} \leq 1 \quad \forall (i,j) \in \mathcal{E} \quad (12)$$

$$\sum_{(i,j) \in \mathcal{E}} f_{ij}^{d,o} + \sum_{i \in \mathcal{V}} f_i^{d,o} \gamma^{pr} \leq H_{d,o} \quad \forall d \in \mathcal{V}, o \in \mathcal{O} \quad (13)$$

$$y_{i,k}, y_{ij,k} \in \{0, 1\} \quad \forall i, (i,j), k \quad (14)$$

$$f_i^o, f_i^{d,o}, f_{ij}^{d,o} \in \{0, 1\} \text{ or } \in [0, 1] \quad \forall i, (i,j), d, o \quad (15)$$

The objective is to minimize the total network cost, taking into account the cost of activating and operating the physical storage, compute, and transport resources allocated to the SDvCDN, and it is computed as a function of the global flows and the set of active resources, as described in (1). Constraints (2) impose flow conservation at each node. The outgoing flow is composed of the transport flow leaving node  $i$  to its outgoing neighbors,  $\delta^+(i)$ , for demand  $(d, o)$ , and the local demand  $q_i^{d,o}$ , which must be equal to the incoming flow, composed of the transport flow entering node  $i$  from its incoming neighbors,  $\delta^-(i)$ , for demand  $(d, o)$ , and the storage flow at node  $i$  for demand  $(d, o)$ . Constraints (3) and (4) model the unicast or multicast nature of storage and transport flows. Since a single stored object can be used to satisfy multiple demands, storage flows are said to be multicast. Hence, user-object storage flows for the same object, but for different destinations, are allowed to overlap. On the other hand, since the use of multicasting cannot be exploited for today's dominant video on-demand services, we assume transport flows to be unicast. As such, user-object transport flows cannot overlap and must be added across both objects and destinations. Note that user-object transport flows must be sized by both the size of the object,  $B_o$ , and the request rate for object  $o$  at destination  $d$ ,  $\lambda_{d,o}$ . Finally, constraints (6) compute the load on the processing resources that prepare the content to be delivered. Content flows must be processed for each demand  $(d, o)$  and hence are unicast flows, but they only get processed once for each  $(d, o)$  at the location where the content is stored. Constraints (7), (8) and (9) are capacity constraints and, constraints (10), (11) and (12) impose that only one capacity level can be activated. Finally, constraints (13) impose a maximum delay for the delivery of each demand  $(d, o)$ . Note that the delay associated to the delivery of object  $o$  for destination  $d$  is computed by adding the user-object flows for demand  $(d, o)$  over all links (for transport delay) and nodes (for processing delay,  $\gamma^{pr}$ ), which essentially computes the number of (transport and processing) hops on the path used to deliver  $o$  to  $d$ . It is important to note that constraint (13) requires the use of binary flow variables. In fact, precisely computing end-to-end delay with fractional flow variables is a hard open problem [17].

We remark that, while the complexity of a MILP solution increases exponentially with the number of integer (binary) variables, in our problem, this number can be significantly reduced based on the following observations. Regarding the flow variables, as shown in [9], if maximum delay constraints are not required, binary flow variables in mixed-cast flow problems can be relaxed to be fractional without violating feasibility if any of the following conditions are satisfied: (1) all flows are unicast (e.g., unicast delivery with fixed stored content sources); (2) the network topology is a tree; (3) the network can perform intra-session network coding (e.g., random linear coding). With respect to the binary resource activation variables, they can be completely eliminated in the case of: (1) linear costs, which is a common assumption in existing content distribution studies, or, as used in lower-bounded facility location problems [18], (2) when capturing resource consolidation savings by lower-bounding the load on each resource. Note that this indicates the existence of regimes of practical interest in which the joint content-resource allocation problem admits solutions of reduced and even polynomial-time complexity.

### 3 Numerical Experiments

This section is devoted to the experimentation of the proposed MILP formulation on a sample network. In order to precisely capture delay constraints and activation costs, we consider the ILP that results from constraining both activation and flow variables to be integer (binary).

#### 3.1 Instance Description

We consider a representative US continental network, composed of core and metro network segments. For the core network segment, we use the Abilene US continental core network [19], comprising 11 nodes and 28 directed links. We assume that each core node has a hierarchical metro network attached to it. We consider a representative metro network with 2 layers: the metro Point of Presence (PoP) layer, with 3 PoP nodes, and the metro edge layer, with 5 edge nodes per PoP. The links between core nodes and PoP nodes, and between PoP and edge nodes are directed in the downstream direction. Since we focus on a content distribution application, we ignore the upstream metro links. Hence, each metro comprises 18 directional links. The network has then a total of 209 nodes and 226 directed links. Edge nodes aggregate wireline or wireless access points and hence are the only nodes in our network generating demand. We then have 165 destination nodes. We assume a future scenario in which the cloud network operator will have DCs hosting cloud resources at each of the 209 (core and metro) network nodes. In the following, we refer to this

network as *Real*. Moreover, in order to show the advantage of creating additional virtual links, we consider the networks resulting from the addition of the following two sets of virtual links:

- *Set 1*: The set of directed links between each core node and every edge node in their respective metros (165 directed links);
- *Set 2*: The set of directed links between any two nodes that are two hops away in the *Real* core network (38 directed links).

The network resulting from the addition of *Set 1* is referred to as *Virtual1*, while the one resulting from the addition of both *Set 1* and *Set 2* is referred to as *Virtual2*. The *Virtual2* network with all the *Set 2* virtual links and a subset of the *Set 1* virtual links for one of the metros is illustrated in Fig. 1 (virtual links are represented in light grey, while black is used for the *Real* network).

In terms of storage, compute, and transport costs and capacities, we consider the following values, derived from real undisclosed US operators' annual capital and operational expenses:

- Storage and processing efficiencies are assumed to be homogeneous across network nodes:  $e_i^{st} = 1$  (\$/GB);  $e_i^{pr} = 100$  (\$/Gbps).
- Transport efficiencies are homogeneous across network segments:  $e_{ij}^{tr} = 300 + 10 l_{ij} / l_{max}$  (\$/Gbps) between core nodes, where  $l_{ij}$  is the length of link  $(i, j)$  and  $l_{max}$  is the maximum link length;  $e_{ij}^{tr} = 200$  (\$/Gbps) between core and metro PoP; and  $e_{ij}^{tr} = 100$  (\$/Gbps) between metro PoP and metro edge. We use  $e_{ij}^{tr} = 250$  (\$/Gbps) for virtual links between a core and a metro edge node, and  $e_{ij}^{tr} = 400$  (\$/Gbps) for virtual links between two core nodes.
- Storage and processing activation costs are homogeneous across network nodes, with two capacity levels, one from 0 to 50% and another one from 50 to 100% of

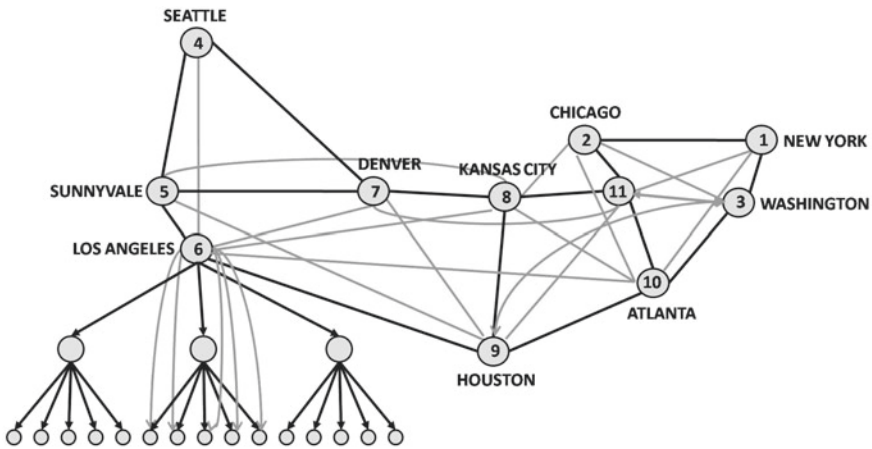


Fig. 1 Illustration of the *Virtual2* network topology

the entire library size  $|\mathcal{O}|B_o$  (for storage), and of the entire demand rate  $|\mathcal{D}|\beta$  (for processing):  $a_{i,1}^{st} = 1000$  \$ and  $a_{i,2}^{st} = 1500$  \$;  $a_{i,1}^{pr} = 10000$  \$;  $a_{i,2}^{pr} = 15000$  \$.

- Transport activation costs are homogeneous across network links with two capacity levels, one from 0 to 50% and another one from 50 to 100% of the entire demand rate  $|\mathcal{D}|\beta$ :  $a_{ij,1}^{tr} = 500$  \$;  $a_{ij,2}^{tr} = 1000$  \$.

Regarding the content to be delivered by the SDvCDN, we consider Netflix content as the most dominant VoD provider. In particular, we consider a total of 50,000 movie titles of size 3 GB each. We assume movie titles are requested according to a Zipf popularity distribution [20] with Zipf shape parameter equal to 0.8 [21]. We classify the 50,000 titles into  $|\mathcal{O}| = 50$  popularity classes. In order to achieve higher granularity for the most popular titles, we set the size of the object popularity classes containing the most popular titles to be the smallest. In particular, we set each popularity class to account for the same aggregate demand, leading to very small groups for the most popular titles and increasingly larger groups for the least popular titles. This allows making finer decisions for the most popular objects that tend to get stored in different levels of the network, while making joint decisions for large groups of unpopular objects that typically end up getting stored together at the core network level. In terms of demand rates, we consider the total metro video traffic, measured in year 2014, and projected for years 2018 and 2022, from a major US network operator, and assume Netflix traffic accounts for 30% of total video traffic [22], leading to the following demand rates per destination (metro edge node),  $\beta_d = \beta, \forall d$ : year 2014,  $\beta = 72$  Gbps; year 2018,  $\beta = 157$  Gbps; year 2022,  $\beta = 313$  Gbps.

Finally, regarding the maximum delay constraints, we assume homogeneous delay constraints  $H_{d,o} = H, \forall d, o$ , with  $H$  ranging from 1 to 3 hops depending on the scenario under investigation. The processing delay is assumed to be equivalent to one transport hop and hence  $\gamma^{pr} = 1$ .

## 3.2 Results

In the following, we present the results obtained by solving the proposed ILP formulation on different problem instances. The goal is to illustrate the effect of the different features captured by our model, including the effect of resource activation costs, delay constraints, and virtual connectivity. All instances were solved using the optimization software CPLEX on an Intel(R) Core(TM) i7-4600M, CPU 2.90 GHz, 8,00 GB RAM. Remarkably, all instances are solved within 15 mins, and most of them within 5 mins, confirming the suitability of our approach for the reconfiguration of SDvCDNs in which significant traffic demand changes due to variations in object popularity happen at the hours time-scale. We present results for 2014, 2018, and 2022 VoD traffic on the following network settings:

- *Real\_nAC\_nH*: Real network with no activation costs, and no delay constraints;
- *Real\_AC\_nH*: Real network with activation costs, and no delay constraints;
- *Real\_AC\_H=3*: Real network with activation variables, and delay  $H = 3$ ;

- *Real\_AC\_H=2*: Real network with activation costs, and delay  $H = 2$ ;
- *Virtual2\_AC\_H=2*: Virtual2 network with activation costs, and delay  $H = 2$ ;

For each year and network instance, Table 1 reports: the number of active nodes at each network level; the three efficiency cost components; the three activation cost components; and the total cost. We make the following observations:

1. When no activation costs are taken into account, the optimal solution activates all network nodes. While not shown in Table 1 due to space limitations, the solution distributes the smaller-size object classes containing the most popular files at the metro edge, and then progressively stores the larger object classes (containing lower popularity files) at the metro PoP and core levels. In particular, for *Real\_nAC\_nH* under 2022 traffic, objects [1–22] are stored at the 165 metro edge nodes, objects [23–45] are stored at the 33 metro PoP nodes, and objects [46–50] are stored at the 11 core nodes.

2. The introduction of the resource activation costs that model the fact that operators have to pay a fixed cost when reaching certain capacity levels, significantly affect the structure of the optimal solution. Indeed, the results for *Real\_AC\_nH* show that no metro edge nodes are activated. In this case, all objects are stored at either metro PoP or core network levels. Note that since no maximum delay constraints are imposed, the solution has the freedom to consolidate resources in order to save costs without worrying about resulting delay penalties.

3. When imposing maximum delay constraints, the optimal solution is now pushed towards higher levels of distribution, specially for some of the objects stored at the core network level that may be violating the delay constraints. This effect is specially visible in *Real\_AC\_H=2*, where all the objects are forced to be stored at the metro PoP level.

4. When we take into account the ability to create virtual links, the model returns solutions where the network can still achieve a good level of resource consolidation, and hence costs savings, while still meeting the delay constraints. In fact, observing the solutions for *Virtual2\_AC\_H=2*, and comparing them with the solutions for *Real\_AC\_H=2*, we can see how in *Virtual2\_AC\_H=2* core network nodes can become active again to consolidate a significant number of object classes, while still meeting the delay constraint by delivering the requested objects via direct virtual links between core and metro edge nodes.

Finally, note that all above observations hold for each traffic demand year. While the trends are similar, observe that the total cost increases with the increase of total traffic demand. In addition, while not shown in Table 1 due to space limitations, the level of distribution also increases with the traffic demand, as higher traffic demands translate into higher transport costs that push content closer to the edge.

**Table 1** Active nodes, cost components, and total costs for the test network instances

		2014						2022					
Instance	# Activated nodes		EDGE	$St_{eff}$	$Pr_{eff}$	$Tr_{eff}$	$St_{ac}$	$Pr_{ac}$	$Tr_{ac}$	Total			
	CORE	POP											
<i>REAL_nAC_nH</i>	11	33	165	1475310	2241591	1188000	-	-	-	4904901			
<i>REAL_AC_nH</i>	11	33	0	1391358	1188000	2552555	46000	44000	104000	5325913			
<i>REAL_AC_H = 3</i>	11	33	0	2006532	1188000	2138338	49500	44000	99000	5525370			
<i>REAL_AC_H = 2</i>	0	33	0	4950000	1188000	1188000	49500	33000	82500	7491000			
<i>VIRTUAL2_AC_H = 2</i>	11	33	0	1879812	1188000	2007660	49500	44000	165000	5333972			
2018													
Instance	# Activated nodes		EDGE	$St_{eff}$	$Pr_{eff}$	$Tr_{eff}$	$St_{ac}$	$Pr_{ac}$	$Tr_{ac}$	TOTAL			
	CORE	POP											
<i>REAL_nAC_nH</i>	11	33	165	2744940	2590500	3056685	-	-	-	8392125			
<i>REAL_AC_nH</i>	11	33	0	2523180	2590500	3937513	49500	44000	99000	9243693			
<i>REAL_AC_H = 3</i>	11	33	0	2523180	2590500	3937513	49500	44000	99000	9243693			
<i>REAL_AC_H = 2</i>	0	33	0	4950000	2590500	2590500	49500	33000	82500	10296000			
<i>VIRTUAL2_AC_H = 2</i>	11	33	0	2256674	2590500	3833915	49500	44000	165000	8939589			
2022													
Instance	# Activated nodes		EDGE	$St_{eff}$	$Pr_{eff}$	$Tr_{eff}$	$St_{ac}$	$Pr_{ac}$	$Tr_{ac}$	TOTAL			
	CORE	POP											
<i>REAL_nAC_nH</i>	11	33	165	4312638	5164500	3924663	-	-	-	13401801			
<i>REAL_AC_nH</i>	11	33	0	3711510	5164500	6197388	60500	44000	99000	15276898			
<i>REAL_AC_H = 3</i>	11	33	0	3711510	5164500	6197388	60500	44000	99000	15276898			
<i>REAL_AC_H = 2</i>	0	33	0	4950000	5164500	5164500	49500	33000	82500	15444000			
<i>VIRTUAL2_AC_H = 2</i>	11	33	0	3169386	5164500	6403980	49500	44000	165000	14996366			

## 4 Conclusions

In this paper, we formulate the joint content-resource allocation problem in SDvCDNs as a minimum cost mixed-cast flow problem that integrates placement, routing, and (virtual) network design decisions in the form of a mixed integer linear program (MILP). Our solution jointly optimizes the placement and routing of content objects along with the allocation of storage, compute, and transport resources, is applicable to arbitrary network topologies, and captures activation and operational costs, popularity settings, unicast and multicast delivery, as well as capacity and latency constraints. We show results for a number of network settings based on the Abilene US continental core network and representative US metro networks that illustrate the advantage of optimizing the allocation of resources in a rich SDvCDN environment. We show that the binary version of the problem can be efficiently solved within minutes for the tested scenarios. Future research developments could try to investigate further facility location concepts and ideas, already used in other fields and consolidated in the location theory [23, 24], to integrate them, if possible, in communication network context.

## References

1. Alcatel-Lucent Strategic White Paper: The Programmable Cloud Network—A Primer on SDN and NFV, June (2013)
2. Weldon, M.: The Future X Network. CRC Press, October (2015)
3. Baev, I.D., Rajaraman, R., Swamy, C.: Approximation algorithms for data placement in arbitrary networks. In: ACM SODA'01 (2001)
4. Borst, S., Gupta, V., Walid, A.: Distributed caching algorithms for content distribution networks. In: IEEE INFOCOM'10. San Diego (2010)
5. Krishnan, P., Raz, D., Shavitt, Y.: The cache location problem. *IEEE/ACM Trans. Netw.* **8**(5), 568–582 (2000)
6. Kalpakis, K., Dasgupta, K., Wolfson, O.: Optimal placement of replicas in trees with read, write, and storage costs. *IEEE Trans. Par. Distr. Sys.* 628–637 (2001)
7. Qiu, L., Padmanabhan, V., Voelker, G.: On the placement of web server replicas. *IEEE INFOCOM'01* **3** (2001)
8. Korupolu, M.R., Dahlin, M.: Coordinated placement and replacement for large-scale distributed caches. *IEEE Trans. Know. Data Eng.* **14**, 1317–1329 (2002)
9. Llorca, J., Tulino, A.M.: The content distribution problem and its complexity classification. In: Alcatel-Lucent Technical Report (2013)
10. Fischer, A., Botero, J., Beck, M., de Meer, H., Hesselbach, X.: Virtual network embedding: a survey. *IEEE Comm. Surv. Tutor.* **15**(4) (2013)
11. Broberg, J., et al.: MetaCDN: harnessing storage clouds for high performance content delivery. *J. Net. Comp. App.* **32** (2009)
12. Srinivasan, S., et al.: ActiveCDN: cloud computing meets content delivery networks. In: Technical Report. Columbia University (2012)
13. Jin, Y., et al.: CoDaaS: an experiment cloud-centric content delivery platform for user-generated contents. In: ICNC'12 (2012)
14. Bolla, R., Lombardo, C., Bruschi, R., Mangialardi, S.: DROPv2: energy efficiency through network function virtualization. In: *IEEE Network*, pp. 26–32. April (2014)

15. Woo, H., Han, S., Heo, E., Kim, J., Shin, S.: A virtualized, programmable content delivery network. services, and engineering. In: *IEEE Mobile Cloud Computing* (2014)
16. Llorca, J., Sterle, C., Tulino, A.M., Choi, N., Sforza, A., Amideo, A.E.: Joint content-resource allocation in software defined virtual CDNs. In: *IEEE ICC'15 CCSNA Workshop*. England, London (2015)
17. Grtschel, M., Raack, C., Werner, A.: Towards optimizing the deployment of optical access networks. *EURO J. Opt. Comput.* **2**, 17–53 (2014)
18. Ahmadian, S., Swamy, C.: Improved approximation guarantees for lower-bounded facility location. *CS arXiv*, September (2012)
19. <http://web.archive.org/web/20080113120821/>; <http://abilene.internet2.edu/>
20. Breslau, L., et al.: Web caching and Zipf-like distributions: evidence and implications. In: *INFOCOM'99*
21. Cha, M., Rodriguez, P., Crowcroft, J., Moon, S., Amatriain, X.: Watching television over an IP network. In: *ACM Internet Measurement Conference (IMC)*. Greece, October (2008)
22. Cisco Visual Networking Index. Forecast and methodology, 2013–2018. Cisco, June (2014)
23. Bruno, G., Genovese, A., Piccolo, C.: Capacity management in public service facility networks: a model, computational tests and a case study. *Opt. Lett.* **10**(5), 975–995 (2016)
24. Barbati, M., Piccolo, C.: Equality measures properties for location problems. *Opt. Lett.* **10**(5), 903–920 (2016)



# Facility Location with Item Storage and Delivery

Stefano Coniglio, Jörg Fliege and Ruth Walton

**Abstract** We discuss part of an ongoing research activity involving the University of Southampton and the Royal British National Lifeboat Institution (RNLI), aimed at improving the RNLI's warehousing and logistics operations. In particular, we consider a facility location problem to determine the optimal number and location of warehouses and which items are to be stored in each of them, minimising the costs of storage and transportation. We propose a mixed-integer non-linear programming formulation for the problem, which we then linearise in two different ways and solve to optimality with CPLEX. Computational results are reported and illustrated.

**Keywords** Facility location · Warehousing · Mixed-integer linear programming

## 1 Introduction

The paper describes a portion of an ongoing research activity carried out by the authors at the University of Southampton in collaboration with the British Royal National Lifeboat Institution (RNLI). The RNLI is the largest life-saving charity active on the coasts of UK, Ireland, Channel Islands and Isle of Man. It currently operates 237 lifeboat stations and more than 350 lifeboats, with an (almost entirely volunteer) crew of 4600 people. They respond to emergency calls from around the country, supporting Her Majesty's Coastguard (the section of the maritime and coastguard agency which coordinates maritime search and rescue operations) and provide a 24 h search and rescue service to any people or vessels in distress.

---

S. Coniglio · J. Fliege · R. Walton (✉)  
University of Southampton, University Road, Southampton SO17 1BJ, UK  
e-mail: r.walton@soton.ac.uk

S. Coniglio  
e-mail: s.coniglio@soton.ac.uk

J. Fliege  
e-mail: j.fliege@soton.ac.uk

The research activity is part of a larger project encompassing an overall review of the RNLI's current warehousing and logistics operations. The RNLI currently operates a single warehouse in the UK, the location of which is under review. *Marine items*, encompassing anything related to the lifesaving operations of the RNLI around the coast, such as parts required for boat maintenance, are currently delivered to the warehouse from *suppliers* (external companies which are based in various locations across the UK). From there, they are transported to lifeboat stations, area support centres and divisional bases (to which we refer, collectively, as *demand points*) by an in-house logistics network.

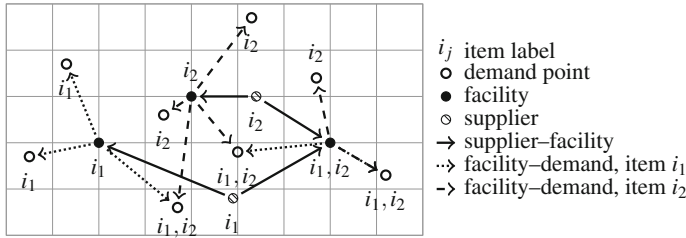
The aim of the paper is to develop optimization techniques to assess the quality of the current warehousing operations at the RNLI and propose improving alternatives by suggesting where to open new warehouses and what to store in them. The work is based on R. Walton's M.Sc. thesis [7].

### The Problem

Given a demand  $c_{ik}$  for each marine item  $i \in I$  at each demand point  $k \in K$ , the road distance  $d_{jk}$  between each potential warehouse (or facility) site  $j \in J$  and demand point  $k \in K$ , as well as the road distance  $e_{ij}$  between the supplier of each item  $i \in I$  and each potential facility site  $j \in J$ , we tackle the problem of deciding how many facilities to open, where to locate them, and which items should be stored in each location, so to meet the demand of marine items at minimum cost. The latter is composed of two parts: that associated with the transportation of goods from suppliers to warehouses and from warehouses to demand points, with a cost per unit distance of transportation equal to  $\alpha$ , and the overhead cost associated with warehouses, namely, a fixed cost  $s_j(p)$  for each warehouse  $j \in J$ , which is decreasing with  $p$ , the total number of warehouses, and the annual cost  $t_i$  for storing any item requiring specialist storage, such as hazardous materials. We categorise the items in  $I$  as *common* (set  $I^C$ ) and *non-common* (set  $I^N$ ), where common items, due to having regular and geographically evenly distributed demand, are stored at all warehouses.

In the problem, each item can be assigned to one or more warehouses, and each demand point is assigned a warehouse from which each individual item is delivered. See Fig. 1 for an illustration. The problem is similar to the *multi-activity facility location* problem proposed in [1], where the decision is not just about where the new facilities should be sited, but also about which items (or *activities*, as they are called in [1]) must be stored (or included) at each facility, so to satisfy the corresponding demand. For more references on facility location problems, we refer the reader to the monography [3].

An additional aspect of the problem we tackle in this paper that takes it away from classical facility location problems is the presence of variable costs  $s_j(p)$ , corresponding to the annual cost of warehousing at site  $j \in J$ , which depends on the total number of warehouses,  $p$ . This is rationalised by the fact that the overhead costs of running one warehouse do not necessarily scale linearly as the number of open warehouses increases. As discussed in the next section, this aspect introduces an element of non-linearity into the problem.



**Fig. 1** Illustration of an instance with two non-common items  $i_1, i_2$ , with facilities to be sited on a grid. Item labels by suppliers, facilities and demand points represent, respectively, items that are offered, stored and required by each of them

Working on the assumption that any new warehouse will not be required to store more items than what is kept in the single warehouse that is currently operated by the RNLI, we assume that storage of an appropriate size is always available for the items assigned to any given location where a warehouse is sited. For this reasons, we do not introduce any warehouse capacity in the problem. It follows that each demand point will be assigned to receive an item from the closest storage location in which that item is stored, as there is no limit on the number of demand points that can be assigned to an individual facility.

**Contributions**

We tackle the problem with Mixed-Integer Linear Programming (MILP) techniques, proposing in Sect. 1 a non-linear formulation which we then linearise in two different ways. We discuss the implementation and results of the formulation to the RNLI case in Sect. 3. Concluding remarks and recommendations for further work are presented in Sect. 4.

**2 Mathematical Programming Formulations**

We start with a mixed-integer non-linear programming formulation for the problem described in the previous section, then followed, after a note on the hardness of the problem, by two alternative linearised formulations. In the first one, the number of warehouses  $p$  is treated as a (general integer) variable. In the second one, we fix  $p$  as a constant—by solving the problem multiple times, for different values of  $p$ , an optimal solution to the original problem is obtained.

**Integer Non-linear Programming Formulation**

Let the integer variable  $p$  be the number of facilities in the solution,  $x_j = 1$  if a facility is located at site  $j$  (and 0 otherwise),  $y_{ij} = 1$  if item  $i \in I$  is stored at location  $j \in J$  (and 0 otherwise),  $z_{ijk} = 1$  if item  $i \in I$  stored at the facility sited at location  $j \in J$  is supplied to demand point  $k \in K$  (and 0 otherwise).

With the definition of  $\alpha$ ,  $c_{ik}$ ,  $e_{ij}$ ,  $d_{jk}$ ,  $s_j(p)$  and  $t_i$  as in the previous section, an integer non-linear programming formulation for the problem is:

$$\min \quad \alpha \left( \sum_{i \in I} \sum_{j \in J} e_{ij} y_{ij} + \sum_{i \in I} \sum_{j \in J} \sum_{k \in K} c_{ik} d_{jk} z_{ijk} \right) + \sum_{j \in J} s_j(p) x_j + \sum_{i \in I} \sum_{j \in J} t_i y_{ij} \quad (1a)$$

$$\text{s.t.} \quad x_j = y_{ij} \quad \forall i \in I^C, j \in J \quad (1b)$$

$$x_j \geq y_{ij} \quad \forall i \in I^N, j \in J \quad (1c)$$

$$y_{ij} \geq z_{ijk} \quad \forall i \in I, j \in J, k \in K \quad (1d)$$

$$\sum_{j \in J} y_{ij} \geq 1 \quad \forall i \in N \quad (1e)$$

$$\sum_{j \in J} z_{ijk} = 1 \quad \forall i \in I, k \in K \quad (1f)$$

$$\sum_{j \in J} x_j = p \quad (1g)$$

$$z_{ijk} = z_{i'jk} \quad \forall i, i' \in I^C, i \neq i', j \in J, k \in K \quad (1h)$$

$$x_j, y_{ij}, z_{ijk} \in \{0, 1\} \quad \forall i \in I, j \in J, k \in K \quad (1i)$$

$$p \in \mathbb{Z}^+. \quad (1j)$$

Constraints (1b) and (1c) ensure that, for every assignment of an item  $i \in I$  to a location  $j \in J$ , a facility is sited at that location. Constraints (1b) impose that common items be stored at every location. Constraints (1c) guarantee that, if an item  $i \in I$  is stored at a location  $j \in J$ , a facility must be open at that location. Similarly, Constraints (1d) ensure that an item  $i \in I$  is supplied to demand point  $k \in K$  from facility site  $j$  if and only if the facility at site  $j$  stores that particular item. Constraints (1e) ensure that all non-common items  $i \in N$  are stored in at least one location  $j \in J$ . This is not required for common items, as the corresponding constraint is superseded by Constraint (1b). Constraints (1f) guarantee that each demand point  $k \in K$  receives each inventory item from exactly one location  $j \in J$ . Constraint (1g) ensures that exactly  $p$  facilities are open, while Constraints (1h) impose that each demand point receives all common inventory items from the same facility. The nature of the decision variables is specified by Constraints (1i) and (1j).

Note the presence of the term  $s_j(p)x_j$ , which is non-linear due to  $x_j$  being a variable and  $s_j(p)$  depending on the variable  $p$ .

## Hardness

We note that the hardness of this problem formalised in (1) is easily established, as it admits the classical Uncapacitated Facility Location Problem (UFLP), which is *NP*-hard in the strong sense [4], as a special case. By setting  $I$  to contain just one common item, the variables  $y_{ij}$  become obsolete, and setting  $e_{ij}, t_i = 0$  for all  $i \in I, j \in J$  and assuming  $s_j(p)$  is a constant, we see that the resulting problem is the UFLP, and as such the problem underlying Formulation (1) is also strongly *NP*-hard.

### MILP Formulation with $p$ as a Variable

Aiming towards a MILP formulation, we eliminate the non-linearity in the objective function as follows. For all  $q \in \{1, \dots, |J|\}$ , let the binary variable  $\mu_q$  be equal to 1 if and only if  $p = q$ . This can be imposed by introducing the constraints  $\sum_{q=1}^{|J|} q\mu_q = p$  and  $\sum_{q=1}^{|J|} \mu_q = 1$ . This way, we can restate the term  $s_j(p)$  as  $\sum_{q=1}^{|J|} s_j(q)\mu_q$ , where  $s_j(q)$  is a constant, and the term  $s_j(p)x_j$  as  $\sum_{q=1}^{|J|} s_j(q)\mu_q x_j$ , thus shifting the non-linearity to  $\mu_q x_j$ . Now let the *continuous* variable  $\lambda_{jq}$  be equal to  $\mu_q x_j$ . Since both  $\mu_q$  and  $x_j$  are binary, we can guarantee  $\lambda_{jq} = \mu_q x_j$  via the following four McCormick envelope constraints [2]:

$$\lambda_{jq} \leq \mu_q \quad \forall j, q \in J \quad (2)$$

$$\lambda_{jq} \leq x_j \quad \forall j, q \in J \quad (3)$$

$$\lambda_{jq} \geq \mu_q + x_j - 1 \quad \forall j, q \in J \quad (4)$$

$$\lambda_{jq} \geq 0 \quad \forall j, q \in J. \quad (5)$$

With those, and the introduction of  $\mu_q \in \{0, 1\}$  for all  $q \in \{1, \dots, |J|\}$ , we can restate the original non-linear term  $\sum_{j \in J} s_j(p)x_j$  in the objective function as  $\sum_{j \in J} \sum_{q=1}^{|J|} s_j(q)\lambda_{jq}$ .

### MILP Formulation with $p$ as a Parameter

An alternative way to circumvent the non-linearity in Formulation (1) is to assume that the number of open warehouses  $p$  is fixed to a given constant. An optimal solution to the unrestricted problem is then obtained by solving the formulation with a fixed  $p$  for each value  $p$  can take, i.e., up to  $|J|$  times.

As better discussed in the next section, there is benefit in being able to compare the solutions that are obtained for different values of  $p$ , as this provides the decision maker with a set of solutions which can be evaluated and compared according to other factors and metrics not accounted for in the original formulation.

## 3 Computational Results

The results that we present in this section have been obtained from an altered version of the original data set provided by the RNLI to prevent sensitive information from being disclosed. The alteration is done in such a way that the final instances are still sufficiently close to the original to allow for computational comparisons, whereas the solutions achieved are not comparable to ensure confidentiality of the results.

The original RNLI instances considered in the project include  $|I| = 24$  items,  $|K| = 233$  demand points and  $|J| = 91$  candidate warehouse sites. The latter are obtained by discretising the UK map on a grid with a granularity of 50 km<sup>2</sup>; this

approach enables us to consider the whole country with a tractable set of potential facility sites, while retaining a sufficiently good geographical precision. To help ensure computational feasibility, the number of items considered is less than the total number of those stored by the RNLI, as including all of them would dramatically increase the problem size. The items that we consider in the experiments are those which an analysis of demand identified as the most representative ones.

Road distances  $e_{ij}$  and  $d_{jk}$  between facility sites, suppliers, and demand points are obtained via Open Source Routing Machine [6], a C++ routing API for use with road networks, based on Open Street Maps [5].

We assume the annual cost per warehouse  $s_j(p)$  to be the same for all  $j \in J$ —the generalisation to the case where this function varies with  $j$  is straightforward. Unlike the warehouse costs, we also consider specialist storage costs  $t_i$  which vary not based on the facility location, but on the nature of the item  $i \in I$  being stored. The value of  $\alpha$  is chosen based on vehicle running costs and fuel cost.

We generate a set of partially randomised instances to provide comparisons with that with RNLI data in terms of computation times. They are generated by randomising the item demands  $c_{ik}$ , as these are the values which are most likely to change when adopting the methods that we propose in future instances of the problem, thus representing a viable method for testing the robustness of the formulation. The values of  $c_{ik}$  are sampled from an exponential distribution fitted to the data provided by the RNLI. We consider instances with increasing  $|I|$  (up to  $|I| = 48$ ). In addition, we test on instances in which the density of the matrix  $C = \{c_{ik}\}_{i \in I, k \in K}$  is increased from 60% (equal to the density in the original RNLI instance) to 100%, to provide further comparisons.

We solve the problem to optimality with CPLEX 12.5, using the two linearised formulations we have introduced (the one with variable  $p$ , and the one with a fixed  $p$ , whose value is set iteratively from 1 to 20). LP relaxations are solved with the `barrier` method after disabling cutting plane generation. The experiments are run on a single node of the University of Southampton's high performance computing cluster IRIDIS (equipped with 16 dual 2.6 GHz Intel Sandybridge processors with 64 GB of RAM), using 6 processors. For comparative purposes, we also solved the MINLP and the MILP formulation with  $p$  as a variable using SCIP 4.0.0. For the only (very small, with  $|I| = |J| = |K| = 10$ ) instance we managed to solve with both formulations, the MINLP one was found to be 7.3 times slower than the linearised alternative. For all the larger instances we tested, SCIP terminated due to exceeding the 16GB RAM limit with a substantially large optimality gap, whereas the linearised formulation was solved to optimality in a matter of minutes.

The results are summarised in Table 1. For larger densities, we register an increase in computation time when employing the formulation with a variable  $p$ , whereas no significant changes occur when adopting the formulation with  $p$  as a fixed parameter. Although both formulations are sensitive to increases in  $|I|$ , with computing times more than doubling from  $|I| = 24$  to  $|I| = 48$ , the formulation with  $p$  as a parameter appears less sensitive to the size of  $|I|$ . In both cases, optimal solutions can still be computed in a reasonable amount of time. We note that with  $p = 1$  CPLEX solves

**Table 1** Computation times with the two MILP formulations, for instances of different size and density

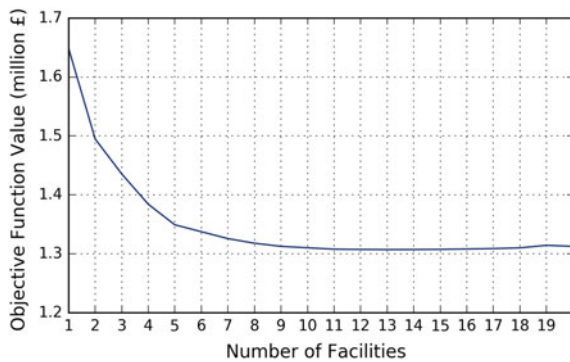
			Variable $p$		Fixed $p$			
					$p = 1$	$p = 2$	$p = 3$	$p = 4$
$ I $	Rndm $c_{ik}$	Dens $c_{ik}(\%)$	$p$	Time	Time	Time	Time	Time
24	No	60	13	1379.0	163.6	61.2	76.9	77.3
24	Yes	60	10	1306.5	161.7	53.7	60.6	58.9
24	Yes	100	15	2713.2	164.5	55.8	60.6	58.0
36	Yes	60	12	3099.7	353.8	85.9	94.5	90.2
48	Yes	60	12	7433.0	631.5	118.2	136.4	126.8

all the instances entirely in presolve, differently from the cases with  $p > 1$  where branch-and-bound is applied after a much shorter presolve phase.

The results of the instance with non-randomised values of  $c_{ik}$  are reported in the first row of the table. For that instance,  $p = 13$  is the value which produces the best result. Only the first four results are shown for the formulation with  $p$  as a parameter, as the computation times remain almost identical for subsequent iterations up to  $p = 20$ . The objective function values obtained on the same instance with the iterated formulation for increasing values of  $p \in \{1, 20\}$  are illustrated in the Fig. 2:

Although a minimum occurs at  $p = 13$ , the objective function curve is reasonably flat around that value, with a steep derivative only for small values of  $p$ —we observe a reduction of 18.2% in the objective function value from  $p = 1$  to  $p = 5$ , and of 3.1% from  $p = 5$  to  $p = 13$ . The total reduction from  $p = 1$  to  $p = 13$  is 20.7%. By looking at the results in more detail, we see that this reduction in cost comes solely from transportation costs. The total cost of warehousing increases with  $p$  due to the additional fixed charge cost with each new warehouse, however the savings in transportation costs overcome the larger warehousing expenses. Indeed, we observe a 65.4% reduction in the total travel distance from  $p = 1$  to  $p = 13$ .

**Fig. 2** Objective function values as a function of  $p \in \{1, 20\}$  on the instance



Note that all the formulations we proposed assume items are all delivered separately, i.e., we consider an individual delivery, per item, from supplier to warehouse and from warehouse to demand point. Since, in reality, the RNLI operates a number of deliveries, each covering more than a single demand point, one should also consider this *vehicle routing* aspect when solving the problem. Since individual deliveries, to which solutions to our formulations are restricted, are clearly a feasible vehicle routing option (although possibly suboptimal), our results provide conservative estimates (i.e., upper bounds) for the generalised problem encompassing vehicle routing aspects. It follows that an overall reduction in the distance travelled using our approach gives a strong indication of a potential for savings when optimal routing has been included.

## 4 Conclusions

We have considered a facility location problem arising in the context of warehouse optimisation within a joint research project involving the University of Southampton and the British Royal National Lifeboat Institution (RNLI). We have proposed two MILP formulations of the problem which allow for variable costs, depending on the number of facilities that have been sited and removing the assumption of identical facilities which is common in facility location problems. We have solved the problem to optimality using both methods, identifying potential savings in transportation costs with the introduction of additional warehouses.

In order to give a more in-depth analysis of potential savings, future work includes the development of solution methods for a joint facility location-vehicle routing problem, where the routing aspect of delivering items from and to any new warehouse site is directly considered.

## References

1. Akinc, U.: Note—multi-activity facility design and location problems. *Manag. Sci.* **31**(3), 275–283 (1985)
2. McCormick, G.P.: Computability of global solutions to factorable nonconvex programs: Part I-convex underestimating problems. *Math. Program.* **10**(1), 147–175 (1976)
3. Mirchandani, P.B., Francis, R.L.: Discrete location theory. In: *Discrete Mathematics and Optimization*. Wiley-Interscience Series. Wiley (1990)
4. Nemhauser, G.L., Wolsey, L.A.: *Integer and Combinatorial Optimization*. Wiley-Interscience (1988)
5. Open Street Map Foundations. OpenStreetMap. <https://master.apis.dev.openstreetmap.org>. Accessed 23 Aug 2016
6. Project OSRM. OSRM backend. <https://github.com/Project-OSRM/osrm-backend/wiki>. Accessed 23 Aug 2016
7. Walton, R.: Warehouse inventory optimisation: a joint facility location inventory approach. Master's thesis. Department of Mathematical Sciences, University of Southampton (2016)



# A Shared Memory Parallel Heuristic Algorithm for the Large-Scale $p$ -Median Problem

Igor Vasilyev and Anton Ushakov

**Abstract** We develop a modified hybrid sequential Lagrangean heuristic for the  $p$ -median problem and its shared memory parallel implementation using the OpenMP interface. The algorithm is based on finding the sequences of lower and upper bounds for the optimal value by use of a Lagrangean relaxation method with a subgradient column generation and a core selection approach in combination with a simulated annealing. The parallel algorithm is implemented using the shared memory (OpenMP) technology. The algorithm is then tested and compared with the most effective modern methods on a set of test instances taken from the literature.

**Keywords**  $p$ -median problem · Parallel computing · OpenMP

## 1 Introduction and Problem Statement

In this short paper we address a famous discrete facility location problem has been in focus of many researchers for more than 50 years. Given a set  $I = \{1, \dots, m\}$  of potential sites for locating  $p \leq m$  facilities, a set  $J = \{1, \dots, n\}$  of customers to be served from open facilities, and  $d_{ij}$  defining the distances (transportation costs) of serving customer  $j \in J$  from the facility  $i \in I$ . The  $p$ -median problem consists in locating  $p$  facilities such that the overall sum of distances from each customer to its closest facility is minimized. Originally, the  $p$ -median problem, first introduced in [9], is to find  $p$  vertices of a weighted graph  $G(I, A)$  such that the sum of weighted distances between the nodes and their closest medians is minimal. In this paper we suppose that  $G(I, A)$  is a simple weighted oriented graph with node set  $|I| = m$ , the arc set  $A = \{(i, j) : i, j \in I; i \neq j\}$ , and the weights  $d_{ij} > 0$  attached to each arc

---

I. Vasilyev (✉) · A. Ushakov

Matrosov Institute for System Dynamics and Control Theory of the Siberian Branch of the Russian Academy of Sciences, 134 Lermontov str., 664033 Irkutsk, Russia  
e-mail: vil@icc.ru

A. Ushakov  
e-mail: aushakov@icc.ru

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_30

$(i, j) \in A$ . To formulate the problem as an integer program, we introduce the standard sets of boolean decision variables  $y_i$ ,  $i \in I$ , and  $x_{ij}$ ,  $(i, j) \in A$ . A variable  $y_i$  indicates whether the node  $i \in I$  is chosen to be a median, while a variable  $x_{ij}$  takes the value 1 if the node  $i$  is a median and  $j$  is assigned to it. Thus, the  $p$ -median problem can be written as follows [3]:

$$\min_{(x,y)} \sum_{(i,j) \in A} d_{ij} x_{ij}, \quad (1)$$

$$\sum_{i \in \delta^-(j)} x_{ij} + y_j = 1, \quad j \in I, \quad (2)$$

$$x_{ij} \leq y_i, \quad i \in I, j \in \delta^+(i), \quad (3)$$

$$\sum_{i \in I} y_i = p, \quad (4)$$

$$y_i, x_{ij} \in \{0, 1\}, \quad i \in I, (i, j) \in A. \quad (5)$$

The objective function (1) is to minimize the overall sum of arc weights between nodes and their closest medians. The constraints (2) ensure that either a node  $j$  is a median or it is assigned to a median. Constraints (3) impose that each node can only be assigned to medians. The constraint (4) enforces that the number of medians must be  $p$ . Let us also introduce the sets  $\delta^-(j) = \{i \in V \mid ij \in A\}$  and  $\delta^+(i) = \{j \in V \mid ij \in A\}$ .

Note that the  $p$ -median problem is proved to be NP-hard on general networks for an arbitrary value of  $p$  [16]. Through the years the  $p$ -median problem has received considerable attention and various both exact and heuristic solution methods have been proposed (for a survey see [5, 18, 20]). The problem has also become popular due to its broad applicability. Maybe one of the most important application of the  $p$ -median problem is clustering [12].

Though the  $p$ -median problem is often able to provide competitive and high quality solutions to the cluster analysis problem [22, 23], they do not often apply it to large scale datasets due to the absence of effective methods of solving such large  $p$ -median problem instances. The most advanced state-of-the-art approaches are able to find exact or good suboptimal solutions to problems on graph with several tens of thousands nodes. Among them are the primal-dual variable neighborhood search [11], branch-and-bound and column generation [6], multistage hybrid algorithm using demand points aggregation and variable neighborhood search [14], a Lagrangean heuristic combined with a data aggregation approach and core heuristic [2]. By now computational results for problem instances with up to 89600 nodes are reported.

There are also parallel algorithms for the  $p$ -median problem including parallel variable neighborhood search [4, 7], a parallel scatter search [8], GPU-based parallel vertex substitution algorithm [17], a parallel cooperative hybridization approach, and a parallelized Lagrangean relaxation-based approach for the discrete ordered median problem [19]. Note that most of the listed approaches are parallelized multi-start techniques. Moreover, there are no any reported computational results for large-scale

problem instances confirming their effectiveness. A distributed parallel algorithm for finding lower bounds for the optimal values of large-scale p-median problem instances was developed in [10].

In this paper we develop an improved modification of the sequential approach proposed in [2] for the p-median problem and its shared memory parallel implementation using the OpenMP interface. The algorithm is based on finding the sequences of lower and upper bounds for the optimal value by use of a Lagrangean relaxation method with a subgradient column generation and a core selection approach in combination with a simulated annealing. The parallel algorithm is implemented using the shared memory (OpenMP) technology. The algorithm is then tested and compared with the most effective modern methods on a set of test instances taken from the literature.

## 2 Sequential Lagrangian Core Heuristic

First of all, let us consider the Lagrangian core heuristic, described in details in [2]. In this short paper we only focus on the main idea of the proposed approach and new features which allows us to improve its efficiency.

First, the Lagrangian dual function is build by relaxing the assignment constraints (2):

$$\mathcal{L}(\lambda) = \min_{(x,y)} \left\{ \sum_{ij \in A} d_{ij} x_{ij} - \sum_{j \in I} \lambda_j \left( \sum_{i \in \delta^-(j)} x_{ij} + y_j - 1 \right) : \text{subject to (3)-(5)} \right\},$$

where  $\lambda = (\lambda_1, \dots, \lambda_n)$  are corresponding Lagrange multipliers. Recall that for any set of  $\lambda$  the Lagrangean dual function provides a lower bound for the objective value.

Let  $\mu_{ij}(\lambda) = d_{ij} - \lambda_j$  and  $\rho_i(\lambda) = \sum_{j \in \delta^+(i)} \mu_{ij}(\lambda) - \lambda_i$  be the reduced cost of the variables  $x_{ij}$  and  $y_j$  respectively. Let  $\rho_i(\lambda)$  be ordered increasingly, i.e. we are given with permutation  $i_1, \dots, i_n$  such that  $\rho_{i_1}(\lambda) \leq \dots \leq \rho_{i_n}(\lambda)$ .

Thus, the value of the Lagrangean dual function is obtained by summing-up the best (i.e. smallest) reduced costs of the variables  $y$  plus the sum of all the multipliers  $\lambda$ , i.e.

$$\mathcal{L}(\lambda) = \sum_{k=1}^p \rho_{i_k}(\lambda) + \sum_{i \in I} \lambda_i,$$

To find the best lower (dual) bound, we form the following dual problem  $\max_{\lambda} \mathcal{L}(\lambda)$ , which is solved with a subgradient algorithm.

Since the efficient computing of  $\mathcal{L}(\lambda)$  and its subgradient requires keeping in RAM the whole ordered distance matrix, we use the approach combining a subgradient algorithm with a delayed column generation and the stabilization technique preventing the oscillation of Lagrange multipliers [11].

To find an upper bound for the optimal value, we apply the well-known core selection approach. Its idea consists in selecting a subset (core set) of “promising” variables and then solving the reduced original problem (1)–(5) over it. Let  $\bar{\lambda}$  be the Lagrange multipliers returned by the subgradient column generation. To determine the core set and formulate the corresponding core problem, we select only the variables whose reduced costs are smaller than given thresholds  $\alpha$  and  $\beta$ , i.e. those variables  $y_i$  and  $x_{ij}$  satisfying

$$i \in I(\bar{\lambda}, \alpha) \triangleq \{i \in I : \rho_i(\bar{\lambda}) \leq \alpha\},$$

$$(i, j) \in W(\bar{\lambda}, \beta) \triangleq \{(i, j) : i \in I(\bar{\lambda}), j \in \delta^+(i), \mu_{ij} \leq \beta\}.$$

In contrast to [2], we developed a parallel implementation of a classical simulated annealing to solve the core problem rather than using a commercial solver. Note that our computational experiments have shown that this approach turns out to be more effective and efficient than solving the core problem with IBM ILOG CPLEX 12.6.3. We adapted a simple cooling rule  $T(t+1) = qT(t)$ , where  $q = (t_0/t_{\min})^{1/(M_{out}-1)}$ ,  $M_{out}$ —a fixed number of temperature reductions,  $t_0$  and  $t_{\min}$  are initial and final temperature respectively.

### 3 OpenMP Implementation Issues

OpenMP interface allows us simply dividing the computation between parallel threads using the shared memory. At the first glance, almost all cycles of our algorithm can be easily parallelized, but some issues to prevent a so-called data race have to be carefully addressed.

Analysing and profiling our code of subgradient optimization and simulating annealing algorithms, we identified the most time consuming parts and bottlenecks which are worth paying close attention:

1. *Distance matrix.* Each column of the distance matrix can be computed and sorted in parallel independently.
2. *Objective function.* The objective function can be computed by running through the sorted columns of the distance matrix from the smallest element to the first median. It can also be done in parallel. For heuristic algorithms, which use a median swapping, several fast computational scheme have been proposed [21, 24]. But it has been our experience that such procedures embedded into our approach are inefficient or even inapplicable in case of large-scale instances, since they require additional amount of RAM for storing auxiliary data.
3. *Dual function.* To compute the value of the dual function, the reduced costs  $\rho(\lambda)$  of  $y$  variables have to be computed and then  $p$  of them with the smallest values have to be taken and summed. To compute  $\rho_i(\lambda)$ , the columns of distance matrix are scanned in parallel. But to avoid a data race, each parallel thread has to keep its own private variables which have then to be summed up. To find  $p$  smallest

reduce cost values of variables  $y$ , we apply the quickselect algorithm [13]. Note that its sequential nature interfered with any our attempts to take advantage of its parallelization, therefore this part of the parallel algorithm remains completely serial.

4. *Subgradient*. Each element of subgradient is computed by scanning the corresponding column of the distance matrix, so there is no any problem for parallelization.

We can conclude that our approach is well adapted for parallelization, but the procedure of computing the value of the Lagrangean dual function reduces the parallel efficiency. This effect will be clearly seen in the results of the computational experiment in the next section.

## 4 Computational Results

In this section we report our preliminary computational results on a subset of large-scale test instances which has previously been used in [2, 11, 14, 15]. The experiments are carried out on the HPC-cluster “Academician V. M. Matrosov” [1]. Our test bed consists of large-scale clustering problems (from 25000 up to 89600 nodes) taken from the literature (e.g. see [11]).

For our parallel subgradient column generation procedure we set the same value of the parameters as in [2]. The parameters  $\alpha$  and  $\beta$  defining the size of the core problem were set so that  $|I(\bar{\lambda}, \alpha)| = 10p$  and  $|W(\bar{\lambda}, \beta)| = 20m$ .

We tested the algorithm using one AMD Opteron 6276 2.3 GHz “Interlagos” processor with 16 cores and 64 GB of RAM available. The quality of the obtained solutions is reported in Table 1, where *Name* is the instance with corresponding  $m$  and  $p$ , *BUB* is the best known upper bound obtained by the approach from [14, 15], *DIV* is the relative difference between the upper bound (*UB*) corresponding to the found solution and *BUB* ( $DIV = \frac{UB - BUB}{BUB} \cdot 100\%$ ), *GAP* is the relative difference between the upper (*UB*) and lower (*LB*) bounds found, i.e. ( $GAP = \frac{UB - LB}{BUB} \cdot 100\%$ ). Note that for the two largest instances *ds1n11* and *ds1n12* with relatively small number of medians  $p$ , we set  $|I(\bar{\lambda}, \alpha)| = 15p$  and  $|W(\bar{\lambda}, \beta)| = 30m$ .

One can observe that the differences between the solutions found with our parallel heuristic algorithm and the best know solutions [14] are negligible. Moreover, it should be pointed out that our parallel approach has an important advantage over metaheuristics like one from [14]: It provides both lower and upper bounds for the optimal value, permitting to estimate the quality of found solutions. In our experiments one can see that the solutions are very close to be optimal.

The parallel simulated annealing allowed us to partially overcome the main drawback of core selection approach in case of small number of medians. In [2] the authors managed the size of core problem such that it can be solved by a commercial solver. On the other hand, our computational experiments demonstrated that the core prob-

**Table 1** Quality of solutions

Name	m	p	BUB	DIV (%)	GAP (%)
ds1n01	25000	25	31229.3	0.000	0.001
ds1n02	36000	36	45115.6	0.000	0.001
ds1n03	49000	49	61384.1	0.001	0.003
ds1n04	64000	64	80053.7	0.001	0.003
ds1n05	30000	25	37563.6	0.000	0.002
ds1n06	43200	36	54191.4	0.000	0.002
ds1n07	58800	49	73626.8	0.000	0.002
ds1n08	76800	64	96039.0	0.001	0.003
ds1n09	35000	25	43902.1	0.000	0.002
ds1n10	50400	36	63169.2	0.000	0.002
ds1n11	68600	49	85833.5	0.001	0.002
ds1n12	89600	64	112059.2	0.008	0.009

**Table 2** Computational time

Name	Time (s)						Speed-up			
	IS	1	2	4	8	16	2	4	8	16
ds1n01	240	556	295	161	96	78	1.9	3.5	5.8	7.1
ds1n02	569	1090	549	299	182	137	2.0	3.6	6.0	7.9
ds1n03	934	1835	934	503	310	225	2.0	3.6	5.9	8.2
ds1n04	1694	2810	1404	762	462	343	2.0	3.7	6.1	8.2
ds1n05	340	780	407	229	137	106	1.9	3.4	5.7	7.3
ds1n06	662	1573	786	431	260	196	2.0	3.7	6.0	8.0
ds1n07	1209	2660	1348	696	425	306	2.0	3.8	6.3	8.7
ds1n08	2305	4099	2092	1079	607	423	2.0	3.8	6.8	9.7
ds1n09	539	1066	546	299	176	132	2.0	3.6	6.0	8.1
ds1n10	984	2267	1111	595	356	249	2.0	3.8	6.4	9.1
ds1n11	1754	3702	2056	974	552	398	1.8	3.8	6.7	9.3
ds1n12	3158	5606	2840	1514	829	590	2.0	3.7	6.8	9.5

lem can be increased and successfully solved with the parallel simulated annealing, providing in some cases better upper bounds.

Table 2 illustrates the computational time (in seconds).  $\mathcal{IS}$  is the running time of Irawan and Salhi’s approach from [14, 15], while the next columns represent the running time and parallel speed-up of our approach with respect to the number of computational cores. The speed-up is computed by  $\text{Speed-up}_i = \frac{T_1}{T_i}$ , where  $T_i$  is the computational time on  $i$  parallel processes. The time  $\mathcal{IS}$  is taken from [15] and non-

malized with respect to one core of our processor using common benchmarks.<sup>1</sup> Such a comparison is certainly not totally fair, but the code of the approach from [15] is not available.

Analysing the presented results we can come to the following conclusions. It is immediately seen that our approach has only one drawback with respect to computational time. But this weakness is overcome by the parallel implementation. Actually, Irawan and Salhi's approach is quite fast, but can be excelled by ours even on two computational cores, while the computational time is significantly reduced up to 16 cores. Moreover, our parallel approach is more general and can be adapted for the different types of data and similarity metrics [23], while Irawan and Salhi's approach, on our opinion, is focused only on well studied two dimensional data on a plane. Observe that the main advantage of our parallel approach, which might be more important than the computational time, is that it finds solutions with tight lower bounds. It allows us to make conclusion on the quality of obtained solutions without prior information (e.g. best known solutions), which is a very valuable feature for practitioners.

We think that highly competitive computational results can be achieved on instances of other types containing several hundreds of thousands nodes without a lot of effort. For larger instances (with several millions nodes) shared memory parallel implementation does not seem to be effective, since its scalability is limited to the number of cores in one processor. Actually, the use of more than 16 cores did not improve the running time even though the processors are on the same board. We assume that the right direction of improving parallel efficiency is to couple the shared memory with the message passing technology. We believe that it allows us to deal with much larger instances on modern high performance computing clusters.

**Acknowledgements** The work of A. Ushakov is supported by the Russian Science Foundation under grant 17-71-10176.

## References

1. Irkutsk Supercomputer Centre of SB RAS (2017). <http://hpc.icc.ru>
2. Avella, P., Boccia, M., Salerno, S., Vasilyev, I.: An aggregation heuristic for large scale  $p$ -median problem. *Comput. Oper. Res.* **39**(7), 1625–1632 (2012)
3. Avella, P., Sassano, A., Vasilyev, I.: Computational study of large-scale  $p$ -median problems. *Math. Program.* **109**(1), 89–114 (2007)
4. Crainic, T.G., Gendreau, M., Hansen, P., Mladenović, N.: Cooperative parallel variable neighborhood search for the  $p$ -median. *J. Heuristics* **10**(3), 293–314 (2004)
5. Daskin, M.S., Maass, K.L.: The  $p$ -median problem. In: Laporte, G., Nickel, S., da Gama, F.S. (eds.) *Location Science*, pp. 21–45. Springer, Cham (2015)
6. García, S., Labbé, M., Marín, A.: Solving large  $p$ -median problems with a radius formulation. *INFORMS J. Comput.* **23**(4), 546–556 (2011)
7. García-López, F., Melián-Batista, B., Moreno-Pérez, J.A., Moreno-Vega, J.M.: The parallel variable neighborhood search for the  $p$ -median problem. *J. Heuristics* **8**(3), 375–388 (2002)

---

<sup>1</sup><http://www.roylongbottom.org.uk>.

8. Garcia-López, F., Melián-Batista, B., Moreno-Pérez, J.A., Moreno-Vega, J.M.: Parallelization of the scatter search for the  $p$ -median problem. *Parallel computing in logistics*. *Parallel Comput.* **29**(5), 575–589 (2003)
9. Hakimi, S.L.: Optimum distribution of switching centers in a communication network and some related graph theoretic problems. *Oper. Res.* **13**(3), 462–475 (1965)
10. Hanafi, S., Sterle, C., Ushakov, A., Vasilyev, I.: A parallel subgradient algorithm for Lagrangean dual function of the  $p$ -median problem. *Studia Informatica Universalis* **9**(3), 105–124 (2011)
11. Hansen, P., Brimberg, J., Urosević, D., Mladenović, N.: Solving large  $p$ -median clustering problems by primal-dual variable neighborhood search. *Data Min. Knowl. Discov.* **19**(3), 351–375 (2009)
12. Hansen, P., Jaumard, B.: Cluster analysis and mathematical programming. *Math. Program.* **79**(1–3), 191–215 (1997)
13. Hoare, C.A.R.: Algorithm 65: Find. *Commun. ACM* **4**(7), 321–322 (1961). doi:[10.1145/366622.366647](https://doi.org/10.1145/366622.366647)
14. Irawan, C.A., Salhi, S.: Solving large  $p$ -median problems by a multistage hybrid approach using demand points aggregation and variable neighbourhood search. *J. Glob. Optim.* **63**(3), 537–554 (2015). doi:[10.1007/s10898-013-0080-z](https://doi.org/10.1007/s10898-013-0080-z)
15. Irawan, C.A., Salhi, S., Scaparra, M.P.: An adaptive multiphase approach for large unconditional and conditional  $p$ -median problems. *Eur. J. Oper. Res.* **237**(2), 590–605 (2014)
16. Kariv, O., Hakimi, S.: An algorithmic approach to network location problems. I: the  $p$ -centers. *SIAM J. Appl. Math.* **37**(3), 513–538 (1979)
17. Ma, L., Lim, G.J.: GPU-based parallel vertex substitution algorithm for the  $p$ -median problem. *Comput. Ind. Eng.* **64**(1), 381–388 (2013)
18. Mladenović, N., Brimberg, J., Hansen, P., Moreno-Pérez, J.: The  $p$ -median problem: a survey of metaheuristic approaches. *Eur. J. Oper. Res.* **179**(3), 927–939 (2007)
19. Redondo, J.L., Marín, A., Ortigosa, P.M.: A parallelized Lagrangean relaxation approach for the discrete ordered median problem. *Ann. Oper. Res.* **246**(1), 253–272 (2014). doi:[10.1007/s10479-014-1744-x](https://doi.org/10.1007/s10479-014-1744-x)
20. Reese, J.: Solution methods for the  $p$ -median problem: an annotated bibliography. *Networks* **28**(3), 125–142 (2006)
21. Resende, M.G.C., Werneck, R.F.: A hybrid heuristic for the  $p$ -median problem. *J. Heuristics* **10**(1), 59–88 (2004)
22. Ushakov, A.V., Klimentova, X., Vasilyev, I.: Bi-level and bi-objective  $p$ -median type problems for integrative clustering: application to analysis of cancer gene-expression and drug-response data. *IEEE-ACM Trans. Comput. Biol. Bioinform.* (2016). doi:[10.1109/TCBB.2016.2622692](https://doi.org/10.1109/TCBB.2016.2622692)
23. Ushakov, A.V., Vasilyev, I.L., Gruzdeva, T.V.: A computational comparison of the  $p$ -median clustering and  $k$ -means. *Int. J. Artif. Intel.* **13**(1), 229–242 (2015)
24. Whitaker, R.A.: A fast algorithm for the greedy interchange for large-scale clustering and median location problems. *Can. J. Oper. Res. Inf. Process.* **21**, 95–108 (1983)



**Part VII**  
**Multi-objective Optimization**

# Robust Plasma Vertical Stabilization in Tokamak Devices via Multi-objective Optimization

Gianmaria De Tommasi, Adriano Mele and Alfredo Pironti

**Abstract** In this paper we present a robust design procedure for plasma vertical stabilization systems in tokamak fusion devices. The proposed approach is based on the solution of a multi-objective optimization problem, whose solution is aimed at obtaining the desired stability margins under different plasma operative scenarios. The effectiveness of the proposed approach is shown by applying it to the *ITER-like* vertical stabilization system recently tested on the EAST tokamak.

**Keywords** Control theory · Robust control · Multi-objective optimization  
Control in nuclear fusion devices

## 1 Introduction

Tokamaks [20] are experimental reactors aimed at proving the feasibility of energy production by means of fusion reaction.<sup>1</sup> In a tokamak, a plasma (a practically fully ionized gas) of hydrogen ions, is confined by magnetic fields and heated to temperatures of the order of several keV (i.e. tens to hundreds millions of degrees). At such high temperatures, collisions between ions can overcome the Coulomb repulsive forces, resulting in fusion reactions. The construction of a tokamak involves the

---

<sup>1</sup>The tokamak concept was first developed in the former Soviet Union in the early 1960s. Indeed, the name tokamak stems from the Russian words for toroidal chamber and magnetic coil, which is *toroidalnaya kamera i magnitnaya katiushka*.

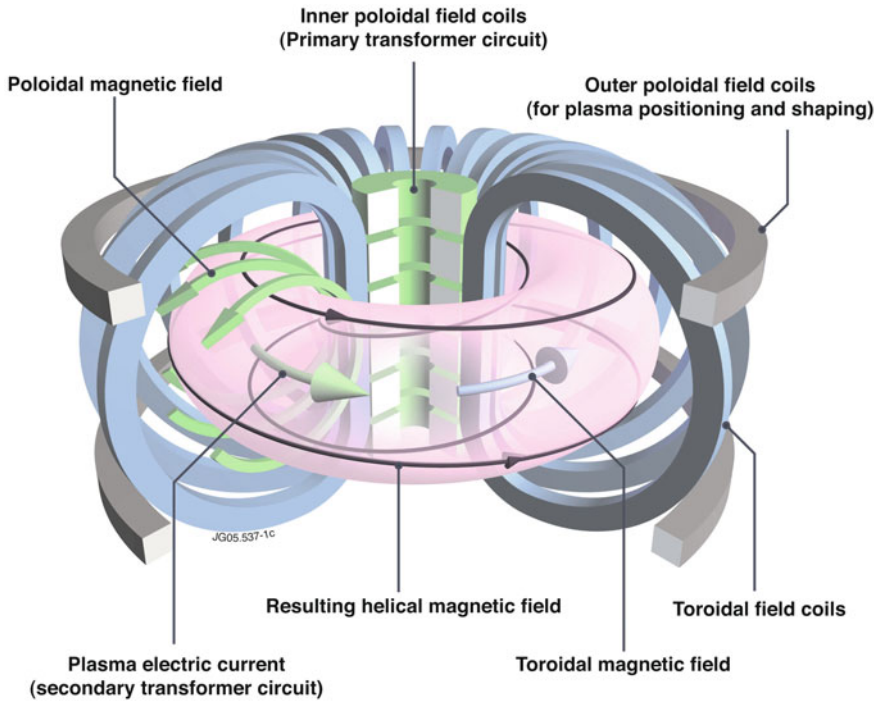
G. De Tommasi (✉) · A. Mele · A. Pironti  
Consorzio CREATE/Dipartimento di Ingegneria Elettrica e delle Tecnologie  
dell'Informazione, Università degli Studi di Napoli Federico II,  
Via Claudio 21, 80125 Naples, Italy  
e-mail: detommas@unina.it

A. Mele  
e-mail: adriano.mele@unina.it

A. Pironti  
e-mail: pironti@unina.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_31



**Fig. 1** Simplified scheme of a tokamak

wrapping a set of toroidal coils around the vacuum vessel to produce a toroidal magnetic field. An additional external poloidal field is needed in order to induce current into the plasma itself, and to change its shape and position. Combining the various components, the net magnetic field lines wind helically around the torus, as shown in the simplified schematic reported in Fig. 1.

Control of the external magnetic field in a tokamak is necessary for a number of reasons; for a complete overview of magnetic control in tokamak devices, the reader is referred to the monograph [5]. Among the various magnetic control systems, the Vertical Stabilization (VS) system is essential to operate tokamaks with elongated and hence vertical unstable plasmas (see [10, Sect. III.A]).

Control algorithms with a simple structure are usually preferred on existing machines when tackling the vertical stabilization problem; indeed a simple structure enables the deployment of effective adaptive algorithms, aimed at robust operations under various scenarios [16, 19]. However, such adaptive algorithms are not always straightforward to be determined; moreover their tuning may require a considerable effort in terms of time, as well as a considerable experience on the specific machine.

Motivated by this fact, an alternative approach for robust vertical stabilization based on multi-objective optimization is presented in this paper. Rather than exploiting online algorithms to adapt the controller gains, robustness is achieved by

solving an offline multi-objective optimization problem. As a case study, the proposed approach is applied to the design of the VS system for the EAST tokamak [21].

The paper is structured as follows: the next section briefly introduces the plasma vertical stabilization problem and the VS control algorithm for EAST originally proposed in [1]. Section 3 presents the main contribution of the paper, i.e. the multi-objective optimization problem exploited to design robust VS system. The effectiveness of the proposed design procedure is illustrated in Sect. 4, by applying it to the design of the EAST VS system. Eventually some conclusive remarks are given.

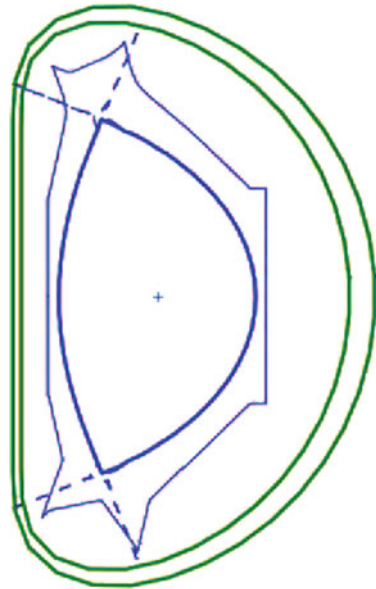
## 2 Plasma Vertical Stabilization

In this section, the plasma vertical stabilization problem is briefly introduced. Then the VS control algorithm considered in Sects. 3 and 4 is presented.

High performance in tokamaks is achieved by plasmas with elongated poloidal cross-section and magnetic X-points, as for the double null plasma shown in Fig. 2 (see also [6, Tutorial 7]). Besides the benefit in terms of fusion performance, the main drawback of such elongated plasmas is that they are vertically unstable (a simplified explanation for plasma vertical instability can be found in [10, Sect. III.A]). It follows that an active stabilization control system is necessary to operate modern tokamaks.

In the current generation of tokamaks with external superconductive coils, an additional pair of in-vessel copper coils are connected in anti-series, in order to form a circuit fed by a single power supply. This circuit generates the required radial magnetic field, on a time scale which is effective to vertically stabilize the

**Fig. 2** EAST pulse #46530 at  $t \cong 5.7$  s. Elongated cross section of a double null plasma



plasma. Therefore, such in-vessel circuit is usually used as actuator by the VS system (among the various tokamak that include the in-vessel coils circuit, there are ITER [3], EAST [21], KSTAR [15] and JT-60SA [8]).

The control algorithms typically adopted for vertical stabilization in existing tokamaks are characterized by a simple structure, which permits to envisage effective online adaptive strategies. The same simple structure can be exploited by the offline approach based on multi-objective optimization proposed in this paper.

In the next sections, we will consider the following VS control algorithm, which was originally proposed for the ITER tokamak in [3], and successfully tested at the EAST tokamak [2]. In particular, the voltage request to the power supply of the in-vessel circuit  $V_{IC}$  is computed as a linear combination of the plasma vertical speed  $v_c$  and of the current flowing in the in-vessel circuit itself; that is the transfer function [12] of the VS control algorithm is

$$V_{IC}(s) = \frac{1 + s\tau_1}{1 + s\tau_2} \cdot \left( K_v \cdot \bar{I}_{p_{ref}} \cdot V_c(s) + K_{IC} \cdot I_{IC}(s) \right), \quad (1)$$

where  $I_{IC}(s)$  is the Laplace transform of the current in the in-vessel coil, while  $\bar{I}_{p_{ref}}$  is the nominal value for the plasma current at each time instant. Moreover, the following parameters of the control algorithm need to be specified:  $K_v$ , which is the *speed gain*<sup>2</sup>;  $K_{IC}$ , which is the *current gain*;  $\tau_1$  and  $\tau_2$ , which are the time constants of the lead compensator [12]. In the next section a procedure based on multi-objective optimization is exploited to set these parameters, in order to achieve robust vertical stabilization in different plasma operational scenarios.

### 3 Robust VS Design via Multi-objective Optimization

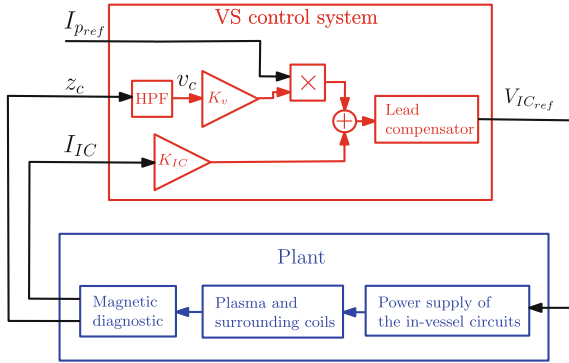
The proposed robust design procedure for the design of VS systems is presented in this section, with reference to the control Algorithm (1). The procedure requires the solution of a multi-objective optimization problem, aiming at maximizing the stability margins ([12, Chap. 6]) for a set of linearized models, which represent the possible plasma scenarios.<sup>3</sup>

In order to compute the stability margins, we consider the single-input-single-output (SISO) transfer function obtained by opening the control loop shown in Fig. 3 in correspondence of the VS control system output.<sup>4</sup> Given the  $i$ -th plasma linearized

<sup>2</sup>The speed gain is multiplied by  $I_{p_{ref}}$  in order to scale the overall gain according to the actual value of the plasma current.

<sup>3</sup>The plasma linear models exploited to solve the optimization problem are generated by the CREATE magnetic equilibrium codes [9].

<sup>4</sup>When computing the open loop transfer function, the simplified model of the power supply of the in-vessel circuit is also considered.



**Fig. 3** Block diagram of the closed loop system considered for the design of the VS system. The plant block includes the power supply of the in-vessel coil, the plasma, the surrounding coils and the so called magnetic diagnostic, which is the system that provides the plant and plasma parameters needed by the magnetic control systems [11, 17]

model, in what follows we will consider an objective function  $\mathcal{F}_i$  that depends on the stability margins of the considered SISO function. Taking into account that the stability margins themselves depend on the parameters of the control algorithm (1), the objective function  $\mathcal{F}_i$  is given by

$$\mathcal{F}_i = c_1 \cdot (PM_t - PM(K_v, K_{IC}, \tau_1, \tau_2))^2 + c_2 \cdot (UGM_t - UGM(K_v, K_{IC}, \tau_1, \tau_2))^2 + c_3 \cdot (LGM_t - LGM(K_v, K_{IC}, \tau_1, \tau_2))^2, \quad (2)$$

where  $PM$  is the phase margin of the SISO open loop function,  $UGM$  and  $LGM$  are the upper and lower gain margins, respectively, while  $c_1, c_2$  and  $c_3$  are positive weighting coefficients.  $PM_t, UGM_t$  and  $LGM_t$  are the desired values (*targets*) for the stability margins.

Given a set of  $N$  linearized models for the plasma and the surrounding coils, the proposed robust design approach requires to minimize  $N$  objective functions (2), one for each plasma equilibrium, exploiting a multi-objective optimization approach. In particular, the following optimization problem is solved

$$\begin{aligned} & \min_{K_v, K_{IC}, \tau_1, \tau_2} \mu \\ & \text{s.t. } \mathcal{F}(K_v, K_{IC}, \tau_1, \tau_2) - \mu \cdot w \leq 0, \end{aligned} \quad (3)$$

where  $\mathcal{F}$  is a vector function

$$\mathcal{F}(K_v, K_{IC}, \tau_1, \tau_2) = (\mathcal{F}_1(K_v, K_{IC}, \tau_1, \tau_2) \dots \mathcal{F}_N(K_v, K_{IC}, \tau_1, \tau_2))^T,$$

and  $w$  is a vector of weights.

Problem (3) can be solved using a sequential quadratic programming method, with modifications to the line search and Hessian [7, 14, 18]. In particular, in the next section, the Matlab® function `fgoalattain` [13] will be exploited to solve (3).

## 4 A Case Study: The ITER-like VS Controller at the EAST Tokamak

In this section we apply the approach described in Sect. 3 to tune the VS algorithm (1) for the EAST tokamak, which is operated in Hefei, by the Institute of Plasma Physics of the Chinese Academy of Sciences.

In particular,  $N = 4$  plasma equilibria have been considered when solving (3). These four equilibria have been chosen so as to represent the possible different operational scenarios for the VS system, in terms of plasma shapes and main parameters (among which we have considered the so called *plasma growth rate*  $\gamma$ , which is the unstable eigenvalue that corresponds to the vertical instability). The main parameters for the considered equilibria are summarized in Table 1. Moreover, the following target values have been used for the stability margins

$$PM_t = 70^\circ, \quad UGM_t = 10 \text{ dB}, \quad LGM_t = 10 \text{ dB},$$

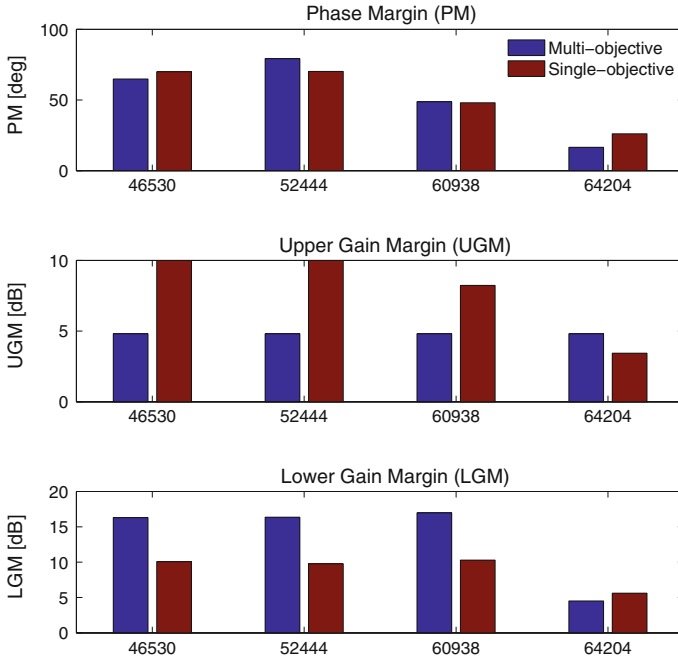
while the coefficients in (2) have been set equal to  $c_1 = 0.1, c_2 = c_3 = 5$ .

The multi-objective problem (3) has been solved using the Matlab® function `fgoalattain`, by setting  $w = (1 \ 1 \ 1 \ 1)^T$ . The results are summarized in Fig. 4, where the values of the stability margins obtained solving (3) are compared with the results obtained solving the single-objective optimization problem

$$\min_{K_v, K_{IC}, \tau_1, \tau_2} \mathcal{F}_i(K_v, K_{IC}, \tau_1, \tau_2), \quad (4)$$

**Table 1** Main plasma parameters of the four EAST equilibria considered to tune the ITER-like VS system (1). The values of the plasma current  $I_{p_{eq}}$  and of the growth rate  $\gamma$  of the vertical instability are reported for each considered equilibrium. Note that each equilibrium is specified by the number of the correspondent EAST pulse and the considered time instant during the pulse itself; these information are reported in the first column of the table

Equilibrium	Shape type	$I_{p_{eq}}$ (kA)	$\gamma$ (s <sup>-1</sup> )
46530 at $t = 3$ s	Double-null	281	137
52444 at $t = 3$ s	Limiter	230	92
60938 at $t = 6$ s	Upper single-null	374	194
64204 at $t = 3.5$ s	Lower single-null	233	512



**Fig. 4** Comparison of the stability margins obtained using the multi-objective approach presented in Sect. 3, and a single-objective approach based on the solution of Sect. 4

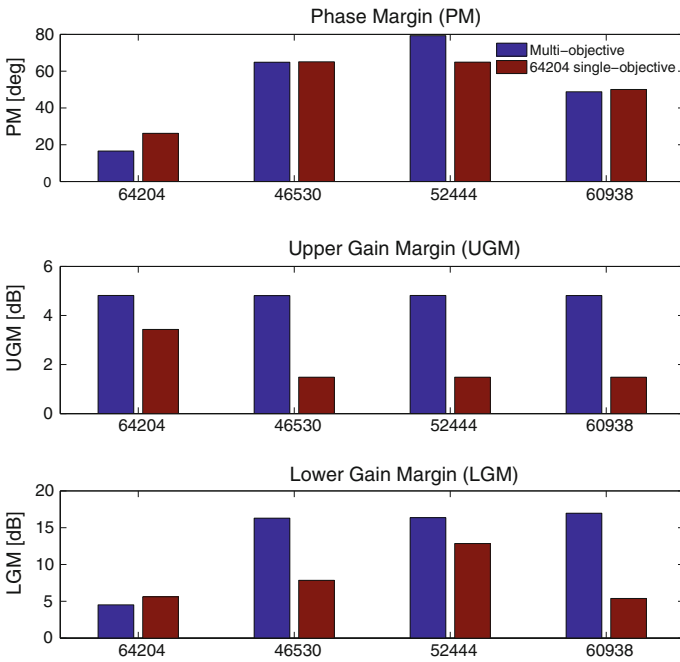
for the  $i$ -th plasma equilibrium, with  $i = 1, \dots, N$ . From the results shown in Fig. 4, it turns out that the multi-objective approach achieves almost the same phase margin  $PM$ , if compared with the single-objective one, while it manages to improve the lower gain margin  $LGM$  at the expenses of the upper gain margin  $UGM$ . In order to better understand the advantage of the multi-objective approach, it should be noticed that, applying to one equilibrium the VS parameters obtained when solving (4) for another one, stability is not guaranteed. Indeed, Table 2 reports the maximum real part of the closed loop eigenvalues computed by using the gains returned when solving (4) for the  $i$ -th equilibrium, to the  $j$ -th one, with  $i \neq j$ . It can be noticed that none of the gains computed on the first three equilibria can stabilize the EAST pulse #64204, which is, among the considered, the one with the largest growth rate. Furthermore, even when stability is achieved, when using the parameters obtained solving (4) on one equilibrium to the other ones, may result in a worsen of the stability margins, as it is shown in Fig. 5. Indeed, by applying the optimal VS parameters—w.r.t problem (4)—obtained for the EAST pulse #64204 to the other equilibria, the gain margins are worst than the ones obtained solving (3).

Finally, if we apply the set of VS gains obtained by solving (3) using the four equilibria reported in Table 1 to the equilibrium of the EAST pulse #70131 at  $t = 4.5$  s,



**Table 2** Maximum real part of the closed loop eigenvalues computed by using the gains returned by the single-objective approach for the  $i$ -th equilibrium, to the  $j$ -th one, with  $i \neq j$

	46530	52444	60938	64204
Single-objective pulse #46530	–	–0.365	–0.088	255.99
Single-objective pulse #52444	–0.360	–	–0.358	897.01
Single-objective pulse #60938	–0.360	–0.364	–	153.57
Single-objective pulse #64204	–0.360	–0.365	–0.358	–



**Fig. 5** Comparison of the stability margins obtained using the multi-objective approach presented in Sect. 3 and by using the VS parameters obtained using the single-objective approach for the EAST pulse #64204

which corresponds to a single-null plasma with  $I_{peq} = 232$  kA and a growth rate  $\gamma \cong 369$  s<sup>-1</sup>, the closed loop system results to be stable with the following margins

$$PM \cong 29.6^\circ, \quad UGM \cong 4.8 \text{ dB}, \quad LGM \cong 11.8 \text{ dB}.$$

## Conclusive Remarks

A robust design procedure for VS systems based on multi-objective optimization has been presented in this paper. The proposed approach can be effectively used to vertically stabilize tokamak plasmas under different scenarios, without the need of online adaptive algorithms. Moreover, differently from other model-based robust approaches, such as [4], the one proposed in this paper, permits to directly specify the requirements in terms of stability margins. The effectiveness of the proposed design procedure has been shown by applying it to the design of the ITER-like VS system deployed at the EAST tokamak.

## References

1. Albanese, R., et al.: A MIMO architecture for integrated control of plasma shape and flux expansion for the EAST tokamak. In: Proceedings of the 2016 IEEE Multi-Conference on Systems and Control, Buenos Aires, Argentina, pp. 611–616 (September 2016)
2. Albanese, R. et al.: ITER-like vertical stabilization system for the EAST Tokamak. *Nucl. Fus.* **57**(8), 086039 (2017)
3. Ambrosino, G., Ariola, M., De Tommasi, G., Pironti, A.: Plasma vertical stabilization in the ITER Tokamak via constrained static output feedback. *IEEE Trans. Contr. Sys. Tech.* **19**(2), 376–381 (2011)
4. Ambrosino, G., Ariola, M., DeTommasi, G., Pironti, A.: Robust vertical control of ITER plasmas via static output feedback. In: Proceedings of the 2011 IEEE Multi-Conference on Systems Control, Denver, Colorado, pp. 276–281 (September 2011)
5. Ariola, M., Pironti, A.: *Magnetic Control of Tokamak Plasmas*. 2nd edition, Springer (2016)
6. Beghi, A., Cenedese, A.: Advances in real-time plasma boundary reconstruction. *IEEE Control Sys. Mag.* **25**(5), 44–64 (2005)
7. Brayton, R. K., Director, S. W., Hachtel, G. D., Vidigal, L.: A new algorithm for statistical circuit design based on quasi-newton methods and function splitting. *IEEE Trans. Circuits Syst, CAS-***26**(9), 784–794 (September 1979)
8. Cruz, N., et al.: Control-oriented tools for the design and validation of the JT-60SA magnetic control system. *Contr. Eng. Pract.* **63**, 81–90 (2017)
9. De Tommasi, G., et al.: XSC Tools: a software suite for tokamak plasma shape control design and validation. *IEEE Trans. Plasma Sci.* **35**(3), 709–723 (2007)
10. De Tommasi, G., et al.: Current, position, and shape control in tokamaks. *Fusion Sci. Technol.* **59**(3), 486–498 (2011)
11. De Tommasi, G., Neto, A.C., Sterle, C.: PIMPA: a tool for optimal measurement probes allocation. *IEEE Trans. Plasma Sci.* **42**(4), 976–983 (2014)
12. Franklin, G. Powell, J. D., Emami-Naeini, A.: *Feedback Control of Dynamic Systems*. Prentice Hall, 5th edn (2006)
13. Gembicki, F. W.: *Vector optimization for control with performance and parameter sensitivity indices*. Ph.D thesis, Case Western Reserve University, Cleveland, OH (1974)
14. Han, S.P.: A globally convergent method for nonlinear programming. *J. Optim. Theory Appl.* **22**(3), 297–309 (1977)
15. Kim, H.K., et al.: Design features of the KSTAR in-vessel control coils. *Fus. Eng. Des.* **84**(2–6), 1029–1032 (2009)
16. Neto, A., et al.: Exploitation of modularity in the JET tokamak vertical stabilization system. *Control Eng. Pract.* **20**(9), 846–856 (2012)
17. Neto, A.C., et al.: Conceptual architecture of the plant system controller for the magnetics diagnostic of the ITER tokamak. *Fus. Eng. Des.* **96–97**, 887–890 (2015)

18. Powell, M. J. D.: Numerical analysis, volume 630 of lecture notes in mathematics, chapter a fast algorithm for nonlinear constrained optimization calculations, pp. 144–157. Springer (1978)
19. Sartori, F., De Tommasi, G., Piccolo, F.: The joint European torus. *IEEE Control Sys. Mag.* **26**(2), 64–78 (2006)
20. Wesson, J.: Tokamaks. Oxford University Press (2004)
21. Yuan, Q.P., et al.: Plasma current, position and shape feedback control on EAST. *Nucl. Fus.* **53**(4), 043009 (2013)

# Performance Analysis of Single and Multi-objective Approaches for the Critical Node Detection Problem

Luca Faramondi, Gabriele Oliva, Roberto Setola, Federica Pascucci, Annunziata Esposito Amideo and Maria Paola Scaparra

**Abstract** Critical infrastructures are network-based systems which are prone to various types of threats (e.g., terroristic or cyber-attacks). Therefore, it is paramount to devise modelling frameworks to assess their ability to withstand external disruptions and to develop protection strategies aimed at improving their robustness and security. In this paper, we compare six modelling approaches for identifying the most critical nodes in infrastructure networks. Three are well-established approaches in the literature, while three are recently proposed frameworks. All the approaches take the perspective of an attacker whose ultimate goal is to inflict maximum damage to a network with minimal effort. Specifically, they assume that a saboteur must decide which nodes to disable so as to disrupt network connectivity as much as possible. The formulations differ in terms of the attacker objectives and connectivity metrics (e.g., trade-off between inflicted damage and attack cost, pair-wise connectivity, size and number of disconnected partitions). We apply the six formulations to the IEEE24

---

L. Faramondi (✉) · G. Oliva · R. Setola  
Università Campus Bio-Medico, Via Alvaro del Portillo 21,  
00128 Rome, Italy  
e-mail: l.faramondi@unicampus.it

G. Oliva  
e-mail: g.oliva@unicampus.it

R. Setola  
e-mail: r.setola@unicampus.it

F. Pascucci  
Department of Engineering, University Roma Tre,  
Via della Vasca Navale 79, 00146 Rome, Italy  
e-mail: pascucci@dia.uniroma3.it

A. Esposito Amideo · M.P. Scaparra  
Kent Business School, University of Kent, Sibson, Park Wood Road,  
Canterbury, Kent CT2 7FS, UK  
e-mail: ae306@kent.ac.uk

M.P. Scaparra  
e-mail: M.P.Scaparra@kent.ac.uk

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_32

and IEEE118 Power Systems and conduct a comparative analysis of the solutions obtained with different parameters settings. Finally, we use frequency analysis to determine the most critical nodes with respect to different attack strategies.

**Keywords** Critical infrastructures · Network vulnerability  
Critical node detection problem

## 1 Introduction

Various recent disasters, either natural (e.g., Katrina hurricane, Chelyabinsk meteor crash), accidental (e.g., US and Italian blackouts) or intentional (e.g., 9/11 terrorist attack, STUXNET cyber attack), have demonstrated that *Critical Infrastructures* (CI) (e.g., power grids, telecommunication networks, etc.,) are highly vulnerable to disruptions. These often lead to complete or partial interruptions of lifeline services provided to communities and, consequently, can have dramatic and even life-threatening consequences. Moreover, due to the high level of interdependency among their sub-components, CIs are prone to cascading failures and localized disruptions can often spread throughout the system with amplified deleterious effects. As an example, the 2003 US blackout was caused by a software glitch into the alarm system of a private energy society in Ohio. The initial local blackout triggered a series of events and ultimately resulted in a widespread outage that affected 8 States and more than 45 million people. To reduce risk and mitigate the aftermath of potential disasters, it is therefore necessary to define modeling approaches to identify the most critical assets and plan protection efforts.

In the next section we briefly overview some of these modeling approaches. Section 3 introduces some notation and definitions. Six formulations for identifying critical nodes are discussed and compared in Sect. 4. An application of the approaches is discussed in Sect. 5, while Sect. 6 offers some conclusive remarks.

## 2 Literature Review

Since the early 2000s, researchers have acknowledged the need to include network topological aspects into modeling approaches aiming at assessing CI vulnerability [1, 2]. The most critical network components, in fact, can be identified by assuming that an attacker, who has knowledge of the network topology, tries to inflict maximum damage with minimal effort by disabling nodes and/or links [3–5]. This type of problems can be formulated mathematically as optimization models, where the component criticality is defined according to different metrics which depend on the specific application domain (e.g., maximal flow, shortest paths and connectivity [6]). In this paper, we focus on critical nodes with respect to connectivity metrics.

A well-known problem in this context is the *Critical Node Detection Problem* (CNP) [7–9]. The CNP assumes that an attacker disables a fixed number of nodes so as to minimize the network *pairwise connectivity* (PWC) (i.e., the number of node pairs that are still connected after the attack) [10]. A dual version of CNP is the  $\beta$ -*Vertex Disruptor* problem [11], which aims at identifying a minimized set of nodes whose removal degrades PWC to a desired degree. A slightly different problem is the *Cardinality Constrained Critical Node Detection Problem* (CC-CNP) [12]. CC-CNP minimizes the number of nodes to be disabled so that the largest connected component is smaller than a user-defined threshold. Another problem variant, the *Component-Cardinality-Constrained Critical Node Problem* (C3-CNP) [13], minimizes the weight of the nodes to be removed so that the total connection costs of each connected component does not exceed a given bound.

It has been argued that critical node detection problems are inherently multi-objective. Major disruptive attack, in fact, could be the one which minimizes different connectivity metrics simultaneously. A multi-objective version of CNP is proposed in [14] to identify the set of  $K$  nodes whose removal maximizes the total number of connected components and minimizes the variance in the cardinality (number of nodes) among the connected components. Other multi-objective formulations have been recently introduced by a subset of this work authors in [15, 16].

### 3 Modelling Preliminaries

Let  $G = \{V, E\}$  denote an undirected *graph* with a finite number  $n$  of nodes  $v_i \in V$  and  $e$  edges  $(v_i, v_j) \in E \subseteq V \times V$ . A *path* over  $G$ , starting at a node  $v_i \in V$  and ending at a node  $v_j \in V$ , is a subset of links in  $E$  that connects  $v_i$  and  $v_j$  without creating loops. The length of a path is its number of edges. A *connected component*  $|V_i|$  of  $G = \{V, E\}$  is the maximal set of nodes  $S \subset V$  such that for each  $v_i \in S$  and  $v_j \in S$ , there exists a path in  $G$  from node  $v_i$  to node  $v_j$ . Let  $\hat{G}_A = \{V, E \setminus E_A\}$  be the *residual graph* obtained from  $G$  by removing all edges  $E_A \subseteq E$  incident to each node in a set  $A \subseteq V$ . Namely,  $A$  represents the set of attacked nodes. When a node is attacked, all its incident edges are disabled. As a result,  $G$  is partitioned into a graph  $\hat{G}_A$  and  $|A|$  isolated nodes. Note that the isolated nodes in  $A$  are non included when counting the number of connected components after an attack.

*Pairwise Connectivity* (PWC) is a useful metric to evaluate network robustness and resilience after the removal of some network components [10]. For a graph  $G = \{V, E\}$ ,  $PWC(G)$  is defined as the number of distinct node pairs connected through a path over  $G$ . Formally:

$$PWC(G) = \frac{1}{2} \sum_{v_i, v_j \in V, v_i \neq v_j} p(v_i, v_j) \in \left[0, \frac{n(n-1)}{2}\right] \quad (1)$$

where  $p(v_i, v_j) = 1$  if the pair  $(v_i, v_j)$  is connected; 0, otherwise. Note that, for undirected graphs, the *PWC* is a function of the size  $|V_i|$  of each *connected component*  $G_i = \{V_i, E_i\}$ . In fact, for each connected component  $G_i$  of  $G$ , there are exactly  $|V_i|(|V_i| - 1)/2$  distinct pairs of connected nodes. Assuming that  $G$  contains  $m$  connected components, we can alternatively express Eq. (1) as

$$PWC(G) = \frac{1}{2} \sum_{i=1}^m |V_i|(|V_i| - 1) \quad (2)$$

A normalized version of *PWC* which only takes values in  $[0, 1]$  is the index *nPWC*, defined as:

$$nPWC(G) = \frac{2PWC(G)}{n(n - 1)} \quad (3)$$

## 4 Critical Node Detection Optimization Approaches

In this section, we review the main features of six different formulations aimed at identifying critical nodes in a network. The six formulations, whose features are summarized in Table 1, are the following:

- (I) *CNP*—Critical Nodes Detection Problem [7];
- (II)  *$\beta$ -Vertex Disruptor* [11];
- (III) *CC-CNP*—Cardinality Constrained Critical Node Detection Problem [17];
- (IV) *MPS Optimization*—Max Partition Size [15];
- (V) *PNS Optimization*—Partition Number and Size [16];
- (VI) *MOO*—Multi-Objective Optimization [16].

Note that with the exception of the MOO formulation, all the other approaches are Integer Linear Programs (ILP). In terms of objectives, the CNP formulation minimizes the PWC by removing a fixed number of nodes  $k$  (a proxy of the attack cost). Both the  *$\beta$ -vertex disruptor* and the CC-CNP minimize the attack cost but differ in terms of constraints: the former limits the PWC, while the latter the largest admissible partition size ( $L$ ). The MPS approach minimizes a linear combination of two sub-objectives: the attack cost and the size of the largest partition in the residual graph  $\hat{G}_A$ , while imposing a bound on the number of connected components,  $m$ . The PNS approach uses the same two sub-objectives as MPS. However, in this model the number of connected components in  $\hat{G}_A$  is also maximized instead of being bounded in a constraints. The resulting objective is therefore a weighted combination of three sub-objectives. Finally, the MOO uses a true multi-objective approach to minimize both the nPWC and the attack cost. The model, which does not include constraints on the other problem features ( $m$  or  $L$ ), identifies all the solutions on the Pareto front.

**Table 1** Main features of the formulations

	CNP	$\beta$ -vertex	CC-CNP	MPS	PNS	MOO
Formulation class	ILP	ILP	ILP	ILP	ILP	Non-linear multi-objective
Number of objectives	1	1	1	1 (2 sub-objectives)	1 (3 sub-objectives)	2
Objective function	$\min(PWC)$	$\min(k)$	$\min(k)$	$\min(\alpha k + (1 - \alpha)L)$	$\min(\alpha_1 k + \alpha_2 L - \alpha_3 m)$	$\min(PWC, k)$
Attack cost ( $k$ )	Bounded	Free	Free	Free	Free	Free
Largest partition size ( $L$ )	Free	Free	Bounded	Free	Free	Free
Partitions number ( $m$ )	Free	Free	Free	Bounded	Free	Free
PWC/nPWC	Free	Bounded	Free	Free	Free	Free
Number of constraints	$O(n^2)$	$O(n^3)$	$O(n^2)$	$O(n^2 e)$	$O(n^2 e)$	Unconstrained
Number of variables	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n)$

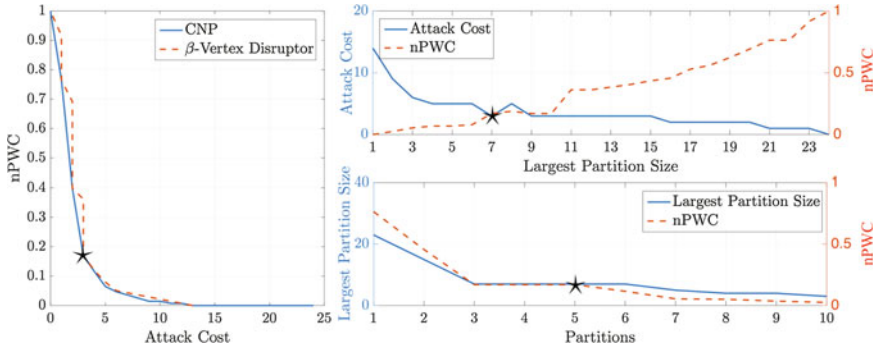
## 5 IEEE-24 and IEEE-118 Power Network Case Study

In this section, we compare the six approaches through their application to the IEEE24 [18] and IEEE118 [19] Power Systems, two networks respectively composed by:  $n = 24$  vertices and  $e = 32$  edges, and  $n = 118$  vertices and  $e = 186$  edges.

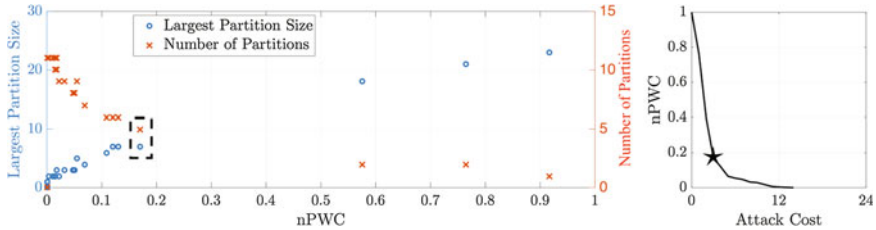
Approaches I–IV handle the intrinsic multi-objective nature of the critical node detection problem by constraining one of the possible degrees of freedom (e.g., attack cost, partitions size, number of partitions, and network connectivity). For these formulations, we explore different solutions obtained by varying the constrained value within a specified range. The first results we report are related to the IEEE24 network. The application of Approach I requires a specific attack cost,  $k$ , that we vary in the range  $[0, n]$ . The plot on the left of Fig. 1a displays the value of the CNP objective function (i.e., the nPWC) for different values of  $k$ . When the attack cost  $k$  increases, the connectivity decreases quite significantly. By removing 5 nodes the connectivity index nPWC drops from 1 to less than 0.1 to denote that the network is highly vulnerable to attacks. The same plot also shows the results of Approach II, which captures the behavior of an attacker aiming at reducing the network connectivity below a threshold value  $\beta$ . For this approach, we consider 21 different values of  $\beta$  in the interval  $[0, 1]$ .

Approach III constraints the size of the largest partition to be smaller than  $L$ . We vary  $L$  in the range  $[1, 24]$ . The plot in the top right corner of Fig. 1a shows that the attack cost (blue solid line) decreases for increasing values of  $L$ . By contrast, the





(a) CNP and  $\beta$ -vertex disruptor comparison varying the attack cost and connectivity (left). CC-CND results varying the largest partition size (top right corner). MPS results varying the number of partitions (bottom right corner).



(b) PNS results for 66 combinations of the objective weights (left). MOO Pareto Front (right)

**Fig. 1** Formulations comparison on the IEEE24 power network

values of  $PWC$  and  $L$  (i.e.,  $\max(|V_i|)$ ) are positively correlated (red dashed line), as implied by Eq. 2.

Approach IV constraints the minimum number of connected components,  $m$ . To generate only realistic attack plans, we vary the value of  $m$  in the range  $[1, 10]$ . In this analysis, we set  $\alpha = 0.8$ . We can observe that, by increasing the number of desired partitions (plot in the bottom right corner of Fig. 1a), both the largest partition size (blue solid line) and the PWC (red dashed line) decrease. This also confirms the relation between the number of partitions, their size, and the network connectivity expressed by Eq. 2.

Approaches V and VI do not include constraints on the solution characteristics. The left-hand side of Fig. 1b shows the results of the PNS approach in terms of nPWC, largest partition size, and number of partitions. The approach was run with 66 different combinations of the objective weights  $\alpha_1$ ,  $\alpha_2$ , and  $\alpha_3$ , so as to reproduce a variety of attack plans (i.e., plans with more emphasis on the attack cost and plans prioritizing other features such as the number and size of the partitions). As expected, the results show that the nPWC and the largest partition size are positively correlated while the nPWC and the number of partitions are negatively correlated.

**Table 2** Most critical nodes in IEEE24 power network

CNP		$\beta$ -vertex		CC-CNP		MPS		PNS		MOO	
<i>id</i>	<i>freq</i> (%)	<i>id</i>	<i>freq</i> (%)	<i>id</i>	<i>freq</i> (%)	<i>id</i>	<i>freq</i> (%)	<i>id</i>	<i>freq</i> (%)	<i>id</i>	<i>freq</i> (%)
16	88	10	71.4	10	60	16	90	9	93.9	10	78.5
13	60	9	47.6	9	48	9	90	16	86.3	16	71.4
11	60	16	33.3	16	44	10	90	10	84.8	9	71.4
24	56	15	23.8	15	32	13	30	11	77.2	21	57.1
20	56	13	14.2	19	28	21	30	24	65.1	11	42.8

Finally, for Approach VI, Fig. 1b shows the 14 solutions on the Pareto front, which capture the relation between the nPWC and the attack cost.

The solutions generated by the six approaches with different parameters form a set of possible attack strategies. To study the similarities of these solutions and identify the most critical nodes with respect to different attack strategies, we compute the frequency of occurrence of each node. Namely, we consider the solutions obtained by applying CNP with 25 different values of parameter  $k$ ,  $\beta$ -vertex with 21 distinct values of nPWC, CC-CNP with 25 values of  $L$ , MPS with 10 values of  $m$ , and PNS with 66 combinations of  $\alpha_1$ ,  $\alpha_2$ , and  $\alpha_3$ . For MOO, we consider the 14 solutions on the Pareto front. Table 2 displays the five most frequent nodes in the solutions obtained with each formulation, along with their frequency.

According to Table 2, it can be noted that the nodes with *id* 9, 10, and 16 are the most critical nodes: they are the most frequently attacked nodes in the solutions of all the approaches, with the exception of CNP.

The graph induced by the deletion of these three nodes is characterized by a nPWC = 0.1703, and is composed by 5 partitions, the largest of which includes 7 nodes. This solution can be obtained by solving CNP with  $k = 3$ ,  $\beta$ -vertex with  $\beta = 0.1703$ , CC-CNP with  $L = 7$ , MPS with  $m = 5$ , and PNS with  $\alpha_1 = 0.7$ ,  $\alpha_2 = 0.2$ , and  $\alpha_3 = 0.1$ . It is also one of the Pareto front solutions of MOO with nPWC = 0.1703 and a cost equal to 3. This particular solution is highlighted in each plot of Fig. 1 by a marker (star) or a dashed box.

Whereas the identification of the three most critical nodes is quite straightforward, the criticality of additional nodes is more difficult to assess as it varies across different models. For example, node 11 is quite critical when formulations CNP, PNS and MOO are used (it appears in the top 5), however it is not as critical according to the attack plans identified by the other approaches.

To have an overall view of the node criticality, in Table 3 we rank all the nodes based on their frequency of occurrence in the 160 solutions generated with all the approaches. Based on this ranking, other highly critical nodes are 21, 24, 13 and 23. In summary, the frequency with which each node appears in the solution sets can be used as a metric of node criticality.

**Table 3** Frequency of occurrence in 160 solutions for each node

<i>id</i>	10	16	9	21	24	13	23	11	2	17	15	20	8	12	5	19	6	4	3	22	14	1	18	7
<i>freq</i> (%)	73.7	71.8	71.8	46.8	41.2	40	38.7	35.6	34.3	33.7	25.6	24.3	21.2	20.6	18.7	16.8	15	12.5	11.8	10	10	10	9.3	5.6

**Table 4** Most critical nodes in IEEE118 power network

CNP		$\beta$ -vertex		CC-CNP		MPS		PNS		MOO	
<i>id</i>	<i>freq</i> (%)	<i>id</i>	<i>freq</i> (%)	<i>id</i>	<i>freq</i> (%)	<i>id</i>	<i>freq</i> (%)	<i>id</i>	<i>freq</i> (%)	<i>id</i>	<i>freq</i> (%)
100	95.6	100	47.6	69	47.8	100	100	100	90.9	49	96.7
80	95.6	49	38	65	43.4	49	100	80	90.9	100	87
49	95.6	69	33.3	49	34.7	37	100	49	89.3	65	82.2
12	91.3	65	33.3	100	30.4	17	100	37	84.8	23	79
92	86.9	23	33.3	80	30.4	77	90	17	83.3	69	72.5
77	86.9	77	23.8	77	30.4	23	90	77	81.8	77	62.9

Table 4 displays the results for the IEEE118 network. Due to the large size of this network, the models were solved with an ant colony-based heuristic with a fixed runtime of 180 s. Overall, 169 sub-optimal solutions were obtained with the following parameter settings:  $k$  and  $L$  vary in the range  $[5, n]$  in steps of 5,  $m$  varies in  $[5, 50]$  in steps of 5; the Pareto front includes 26 solutions;  $\beta$ ,  $\alpha_1$ ,  $\alpha_2$ , and  $\alpha_3$  are set as for the IEEE24 network. As expected, the solutions for this larger network are more dissimilar. Nevertheless, it is still possible to identify some highly critical nodes (e.g. 100, 49, and 77), which recur quite frequently in the solutions.

## 6 Conclusions and Future Work

In this work we reviewed several formulations used to identify the most critical nodes in a network. Each formulation mimics the behaviour of an attacker who aims at disrupting the network connectivity by disabling some critical nodes. To analyse the differences of the solutions obtained when considering different attacker behaviours, we applied six formulations with different parameter settings to the IEEE24 and IEEE118 Power networks. The results suggest that there is a small core set of high critical nodes which consistently appear in the solutions irrespectively of the attacker preferences. However, the remaining set of selected nodes varies depending on the connectivity features and attack cost used by the model. Future research should investigate the integration of these attack models into multi-level optimization approaches to identify cost-effective protection strategies and increase network resiliency to external disruptions.

## References

1. Réka, A., Hawoong, J., Barabási, A.: Error and attack tolerance of complex networks. *Nature* **406**(6794), 378–382 (2000)
2. Holme, P., Kim, B.J., Yoon, C.N., Han, S.K.: Attack vulnerability of complex networks. *Phys. Rev. E* **65**(5), 056109 (2002)
3. Wu, J., Deng, H.Z., Tan, Y.J., Zhu, D.Z.: Vulnerability of complex networks under intentional attack with incomplete information. *J. Phys. A: Math. Theor.* **40**(11), 2665 (2007)
4. Huang, X., Gao, J., Buldyrev, S.V., Havlin, S., Stanley, H.E.: Robustness of interdependent networks under targeted attack. *Phys. Rev. E* **83**, 065101 (2011)
5. Shao, S., Huang, X., Stanley, H.E., Havlin, S.: Percolation of localized attack on complex networks. *New J. Phys.* **17**(2), 023049 (2015)
6. Murray, A.T.: An overview of network vulnerability modeling approaches. *GeoJournal* **78**, 209–221 (2013)
7. Arulselman, A., Commander, C.W., Eleftheriadou, L., Pardalos, P.M.: Detecting critical nodes in sparse graphs. *Comput. Oper. Res.* **36**(7), 2193–2200 (2009)
8. Shen, Y., Nguyen, N.P., Xuan, Y., Thai, M.T.: On the discovery of critical links and nodes for assessing network vulnerability. *IEEE/ACM Trans. Netw. (TON)* **21**(3), 963–973 (2013)
9. Di Summa, M., Grosso, A., Locatelli, M.: Branch and cut algorithms for detecting critical nodes in undirected graphs. *Comput. Optim. Appl.* **53**, 649–680 (2013)
10. Sun, F., Shayman, M.A.: On pairwise connectivity of wireless multihop networks. *Int. J. Secur. Netw.* **2**(1–2), 37–49 (2007)
11. Dinh, T.N., Xuan, Y., Thai, M.T., Park, E.K., Znati, T.: On approximation of new optimization methods for assessing network vulnerability. *INFOCOM* **2010**, 1–9 (2010)
12. Pullan, W.: Heuristic identification of critical nodes in sparse real-world graphs. *J. Heuristics* **21**(5), 577–598 (2015)
13. Lalou, M., Tahraoui, M.A., Kheddouci, H.: Component-cardinality-constrained critical node problem in graphs. *Discret. Appl. Math.* **210**, 150–163 (2015)
14. Ventresca M., Harrison K.R., Ombuki-Berman B.M.: An experimental evaluation of multi-objective evolutionary algorithms for detecting critical nodes in complex networks. In: Mora, A., Squillero, G. (eds.) *Applications of Evolutionary Computation. EvoApplications 2015. Lecture Notes in Computer Science*, vol. 9028 (2015)
15. Faramondi, L., Oliva, G., Pascucci, F., Panzieri, S., Setola, R.: Critical node detection based on attacker preferences. In: *24th Mediterranean Conference on Control and Automation (MED)*, pp. 773–778 (2016)
16. Faramondi, L.: Critical node detection problem: vulnerabilities and robustness metrics. Doctoral dissertation (2017)
17. Arulselman, A., Commander, C.W., Shylo, O., Pardalos, P.M.: Cardinality-constrained critical node detection problem. In: *Performance Models and Risk Management in Communications Systems*, pp. 79–91 (2011)
18. Dam, Q.B., Meliopoulos, A.S., Heydt, G.T., Bose, A.: A breaker-oriented, three-phase IEEE 24-substation test system. *IEEE Trans. Power Syst.* **25**(1), 59–67 (2010)
19. Christie, R.D.: IEEE power systems test case archive, <http://ee.washington.edu/research/pstca>, (1999). Accessed 8 August 2017

# Stable Matching with Multi-objectives: A Goal Programming Approach

Mangesh Gharote, Nitin Phuke, Rahul Patil and Sachin Lodha

**Abstract** Stability in matching has been well studied in the literature. In practice, there are many matching applications where along with stability, other measures such as equity, welfare, costs, etc. are also important. We propose a goal programming based approach for finding a Stable Matching (SM) solution with multi-objectives such as equity and welfare. The goal values are obtained by solving the linear assignment models. On the comparison with prior art, the results of our experiment shows comparable and in many cases significant improvement in the solution quality.

**Keywords** Stable matching · Multi-objective · Optimization · Goal programming

## 1 Introduction

In a matching problem, there are two set of agents (say  $n$  men and  $n$  women). Each man ranks the women in order of his preferences, and each woman ranks the men in order of her preferences. The matching is said to be *unstable*, if there exists any two agents who are not matched to each other but strictly prefer the other to their partner. Gale and Shapley (1962) proposed a Deferred Acceptance (DA) polynomial algorithm for Stable Matching (SM) problem. The DA algorithm produces a man-optimal

---

M. Gharote (✉) · S. Lodha  
TCS Research, Tata Consultancy Services, Pune, India  
e-mail: mangesh.g@tcs.com

S. Lodha  
e-mail: sachin.lodha@tcs.com

N. Phuke (✉)  
Department Computer Engineering & Information Technology,  
College of Engineering, Pune, India  
e-mail: phukeng15.comp@coep.ac.in

R. Patil  
SJM School of Management, Indian Institute of Technology, Bombay, India  
e-mail: rahul.patil@iitb.ac.in

solution, if man proposes first or a woman-optimal solution, if woman proposes first. Thus, the algorithm produces a gender biased matching solution. Since then SM has been widely studied and used in many applications, such as matching of medical students to hospitals [1], students-school admission [2], roommates matching [3], worker-task matching [4], trainee-project matching [5], etc.

In many applications, along with stability other matching measures, like overall preferences matching (welfare), fairness in matching (equity), cost of allocation are also important [6]. For example in an employee-to-project stable matching problem studied by [7], the cost of matching, average preferences satisfaction and stability are all important criteria that need to be considered. Other example, in the marriage problem, the solution should be gender-neutral as well as the agents should get the best match.

Many variants of the SM problem (such as SM with ties in preferences and incomplete list) has been addressed in the literature, but SM with multi-objectives problem is less explored. [8] addressed this problem using evolutionary algorithm and reported the need for an efficient method for solving this problem. The issue of multi-objective matching is addressed in the classical Assignment Problem (AP) literature [9]. AP can be solved efficiently because of the total unimodularity of its coefficient matrix. But with the addition of objective constraints in Goal Programming, it loses this property. The main difference between the AP and SM is the stability criteria [10]. The solutions produced by the assignment problem are not stable. Finding a SM solution with optimum fairness (equity) and social welfare score is computationally challenging [11]. In this paper, we address the SM with multi-objective problem where the preferences of the agents are complete and strictly ordered. The main contributions are as follows:

- A Goal Programming (GP) based approach for solving the SM with multi-objective problem.
- Pareto-optimal solution for SM with equity and welfare as objectives.
- Substantial reduction in the computational time compared to evolutionary algorithm [8].

## 2 Related Work

Gale and Shapley [2] showed that stable marriages always exist and proposed Deferred Acceptance  $O(n^2)$  algorithm to find stable marriages. The algorithm produces gender-biased (man or woman optimal) matching solutions, according to who proposes first in the DA algorithm. To overcome this issue of gender-bias, many new approaches have been proposed. Irving et al. [12] reported an  $O(n^4)$  algorithm for finding optimal SM, with the objective criteria of maximum preferences matching. Vate (1989) showed how to solve *optimal* SM problem as a linear program. The mathematical model comprised of basic assignment constraints (one-to-one matching) and stability constraint which ensures that blocking pairs are not formed in the matching solution.

Many other variants of SM with different objectives have been studied [13]. The problem of finding the welfare-optimal, minimal regret, and fairness optimal stable matching becomes NP-hard and sometimes even hard to approximate [11]. Many researchers focused on local search approaches such as *neighborhood search* [14] for finding near optimal solutions. Also *evolutionary* based methods, mostly genetic algorithm have been studied to obtain SM solution [8]. These approaches are not scalable and often do not produce stable matching.

In this paper, we designed a Goal Programming (GP) based approach to solve the SM Multi-Objective (SMMO) problem. The key idea in using GP is to find solutions which attain a pre-defined target for equity and social welfare function. If there exists no solution which achieves targets in both functional criteria, the task is to find *stable* solutions which minimize deviations from targets [9]. The results of experimentation are compared using Equity and Social Welfare, the functional criteria as discussed in [8]. We also compared our results with the DA algorithm [2], Neighbourhood Search (NS) algorithm [14] and Linear Programming for SM by [15], which are not addressing SMMO problem but can produce quality results compared to [8].

**Deferred Acceptance Algorithm (DA)** The DA-Man proposal algorithm proceeds in rounds. In each round, the man makes a proposal to the most preferred woman who has not yet rejected him. Each woman retains the proposal of the most preferred man she has received (if any), and rejects the rest. The algorithm continues till all men are matched. As the man is proposing first, the solution obtained is biased towards man and is man-optimal SM. Similarly, if woman proposes first then we obtain woman-optimal SM. Thus [2] produces extreme stable marriages.

**Linear Programming for SM** [15] proposed linear program for SM. Here, we describe a mathematical formulation [15] that can be used to find a stable solution.

Notations:

- $N_m$  Number of men  $m = 1, \dots, N_m$
- $N_w$  Number of women  $w = 1, \dots, N_w$
- $j <_m w$  Man  $m$  prefers woman  $w$  over woman  $j$
- $i <_w m$  Woman  $w$  prefers man  $m$  over man  $i$
- $X_{mw} = 1$ , if  $m$  is assigned to  $w$ , otherwise 0.

$$\sum_m^{N_m} X_{mw} = 1 \quad \forall w = 1, \dots, N_w \tag{1}$$

$$\sum_w^{N_w} X_{mw} = 1 \quad \forall m = 1, \dots, N_m \tag{2}$$

$$\sum_{j <_m w, j \neq w}^{N_w} X_{mj} + \sum_{i <_w m, i \neq m}^{N_m} X_{iw} + X_{mw} \leq 1 \quad \forall m = 1, \dots, N_m, \forall w = 1, \dots, N_w \tag{3}$$

$$X_{mw} \in \{0, 1\} \quad \forall m, w; \tag{4}$$



### 3 Problem Statement

Given a complete and strictly ordered preference list of  $n$  men and  $n$  women. The task is to find a stable matching, which maximally meets the objectives. Here we have considered Equity (fairness) and Social Welfare (better preferences) as the objectives. The definitions are as follows: Let,

$r_m(w)$  denotes rank of woman  $w$  in man  $m$  preference list  
 $r_w(m)$  denotes rank of man  $m$  in woman  $w$  preference list

**Equity (E)** determines the fairness in preferences matching. For example, in a Matching  $M$ ,  $r_m(w) = 2$  and  $r_w(m) = 6$ . The equity score is  $|r_m(w) - r_w(m)| = |6 - 2| = 4$ .

**Social Welfare (W)** determines better preferences matching. For the above example, social welfare score is  $\{r_m(w) + r_w(m)\} = 6 + 2 = 8$ .  
 Note, as ranks are ordered from 1 to  $n$ , a rank of one means best preference. Lower score for both E and W is desired.

#### Our Solution Approach

- Obtain the target values for GP by solving  $k$  linear assignment problems, each having a different single objective, where  $k$  is number of objectives. We have considered Equity ( $E_{opt}$ ) and Welfare ( $W_{opt}$ ) as two linear objectives ( $k = 2$ ).

$$E_{opt} = \sum_m^{N_m} \sum_w^{N_w} |r_m(w) - r_w(m)| * X_{mw} \quad (5)$$

$$W_{opt} = \sum_m^{N_m} \sum_w^{N_w} \{r_m(w) + r_w(m)\} * X_{mw} \quad (6)$$

- Solve the SM problem as a weighted GP optimization problem.
- Generate multiple (Pareto-optimal) solutions by varying weights in the objective function.

#### Goal Programming

Given parameters: Weights  $\omega_1$  and  $\omega_2$ ;

Optimal Equity  $E_{opt}$  and Social Welfare  $W_{opt}$  Score

Decision variables:  $X_{mw} = 1$ , if  $m$  is assigned to  $w$ , otherwise 0.

Deviations  $\rho_1$  and  $\rho_2$

Minimize

$$\omega_1 \rho_1 + \omega_2 \rho_2 \tag{7}$$

Subject to

$$\sum_m^{N_m} X_{mw} = 1 \quad \forall w = 1, \dots, N_w \tag{8}$$

$$\sum_w^{N_w} X_{mw} = 1 \quad \forall m = 1, \dots, N_m \tag{9}$$

$$\sum_{j <_m w, j \neq w}^{N_w} X_{mj} + \sum_{i <_w m, i \neq m}^{N_m} X_{iw} + X_{mw} \leq 1 \quad \forall m = 1, \dots, N_m, \forall w = 1, \dots, N_w \tag{10}$$

$$\sum_m^{N_m} \sum_w^{N_w} |r_m(w) - r_w(m)| * X_{mw} - \rho_1 = E_{opt} \tag{11}$$

$$\sum_m^{N_m} \sum_w^{N_w} \{r_m(w) + r_w(m)\} * X_{mw} - \rho_2 = W_{opt} \tag{12}$$

$$X_{mw} \in \{0, 1\} \quad \forall m, w; \quad \rho_i \geq 0 \quad \forall i \tag{13}$$

The objective is to minimize the undesirable deviation variables. Equations 8 and 9 are one-to-one assignment constraints. Equation 10 is a stability constraint which ensures that the unstable pairs are not produced [15]. Here,  $\sum_{j <_m w}^N X_{mj} = 1$  means, man  $m$  is matched to the woman  $j$  that is ranked lower than the woman  $w$  in the man’s preference list. For the solution to be stable  $X_{mw} = 1$ , then  $\sum_{i <_w m, i \neq m}^N X_{iw} = 0$  and  $\sum_{j <_m w, j \neq w}^N X_{mj} = 0$ .

As stated earlier, target values  $E_{opt}$  and  $W_{opt}$  are obtained by solving linear assignment problems (Eqs. 5 and 6) without stability constraints. The solution generated using GP will always lie within  $E_{opt}$  and  $W_{opt}$  scores. To obtain SM solution with lower E and W scores, the undesirable negative deviation variables ( $\rho_1, \rho_2$ ) are added in the Eqs. 11 and 12. Hence, in the objective function the undesirable deviation variables are minimized Eq. 7.

### 3.1 Illustration

We illustrate with a matching problem of size 8 (men = women). The preference lists shown in Table 1 is taken from [12] paper. In the example,  $m_1$  ranks  $w_3$  as his first preference, while  $w_1$  ranks  $m_4$  as her first preference. The numbers in bold indicate the matching solution obtained using our approach (GP). Table 2 displays SM solutions obtained using various methods. Avg. Rank-Men denotes, the average

**Table 1** Preferences men and women

Rank→	1st	2nd	3rd	4th	5th	6th	7th	8th		1st	2nd	3rd	4th	5th	6th	7th	8th
$m_1$	3	1	<b>5</b>	7	4	2	8	6	$w_1$	4	3	<b>8</b>	1	2	5	7	6
$m_2$	6	1	3	<b>4</b>	8	7	5	2	$w_2$	3	7	<b>5</b>	8	6	4	1	2
$m_3$	7	4	<b>3</b>	6	5	1	2	8	$w_3$	7	5	8	<b>3</b>	6	2	1	4
$m_4$	5	3	<b>8</b>	2	6	1	4	7	$w_4$	6	4	<b>2</b>	7	3	1	5	8
$m_5$	4	1	<b>2</b>	8	7	3	6	5	$w_5$	8	7	<b>1</b>	5	6	4	3	2
$m_6$	6	2	5	<b>7</b>	8	4	3	1	$w_6$	5	4	<b>7</b>	6	2	8	3	1
$m_7$	7	8	1	<b>6</b>	2	3	4	5	$w_7$	1	4	5	<b>6</b>	2	8	3	7
$m_8$	2	6	7	<b>1</b>	8	3	4	5	$w_8$	2	5	<b>4</b>	3	7	8	1	6

**Table 2** Illustration results–stable matchings and measures

Method	$m_1$	$m_2$	$m_3$	$m_4$	$m_5$	$m_6$	$m_7$	$m_8$	E	W	Avg. rank-men	Avg. rank-women
GS-Male	3	1	7	5	4	6	8	2	35	55	1.25	5.63
GS-Female	7	8	2	1	6	4	3	5	41	57	6.12	1.00
Vande-Vate	1	3	7	8	4	5	6	2	23	57	2.25	4.87
Our approach	5	4	3	8	2	7	6	1	4	54	3.50	3.25

preference rank of men, while Avg. Rank-Women denotes the average preference rank of women. As discussed, GS-Male proposal DA algorithm produces Male-optimal solution (1.25), while women gets worst preferences (5.63). Similarly, GS-Female results in better preferences matching for women (1.0) and worst for men (6.12). Vande-Vate results in a fair stable solution compared to Gale-Shapley. Vate solves linear program without considering any specific objective function. Using our approach we get best E-Equity and W-Welfare matching stable solution compared to other approaches. The target values ( $E_{opt} = 4$  and  $W_{opt} = 49$ ) were first obtained, by solving linear assignment problem. Then GP SM problem (Eqs. 7–13) is solved using the weights ( $\omega_1 = \omega_2 = 1$ ).

### 4 Results and Discussion

**Experimental Settings** We tested our approach on randomly generated synthetic data sets. The data set comprised of complete and strictly ordered preference list of each man and woman. The generated preferences are uncorrelated as per the experimentation set-up considered in [8]. The results are displayed on six different problems, the problem size varying in the range of 50 to 430. The problem size is defined by the number of men and the equal number of women. We could run the compu-

tations up to 430 size matching problem, due to memory limitation. In GP, weights are varied in the range of  $[0-2]$  with step size of 0.2, where  $\omega_1 + \omega_2 = 2$ . For each fixed weight, model Eqs. (7)–(13) was executed. Such 10 trials for each problem instance were conducted to obtain the Pareto-optimal solutions. All the experiments were carried out on Intel(R) Core i5-4590T CPU 2GHZ processor with 4 GB RAM. In the following section, we discuss our experimentation results.

**Our Approach versus Local Search Approaches** [8] addressed SM with multi-objectives problem using evolutionary based method. They considered objective function criteria as equity and welfare. We implemented their genetic algorithm and experimented with the given parameters (mutation rate = 0.4, crossover rate = 0.6, population size = 50, generations = 5000 and repetitions = 100) as stated in [8] paper. The fitness function is computed in terms of blocking pairs, where lower the fitness value better is the solution. [8] reported results for very small size (20) problem set and in many instances could not find stable solutions. We tested GA for 20 and 100 size problem. Out of 25 replications of problem size 20, on average 5 solutions were found. In case of 100 size problem, GA could not find stable solution even with 20,000 generations. While, with our approach, we could solve large size matching problem and always provide a *stable* solution.

[14] proposed variants of neighborhood search (NS) algorithms for different SM problems such as ties and incomplete preference list. They have not looked at the multi-objective SM problem. But we found an algorithm (SML2) proposed [14] for SM with strict preferences matching quite relevant to the problem addressed. We tested on problem size of P50 and P100, where NS algorithm could not find stable solution even in 1000 steps (773 s), but the number of blocking pairs are significantly less than GA. Thus, NS algorithm is effective than Genetic Algorithm (GA) [8] because it exploits the problem structure while in GA there is no criterion for making the solution stable.

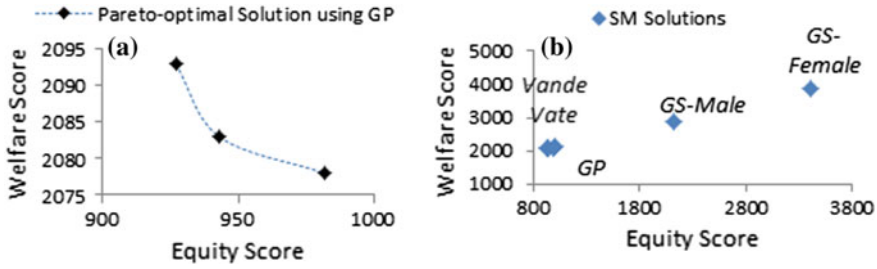
**Our Approach versus Classical Methods** In this section, we compare our results with the existing methods which provide stable solutions, but do not address the multi-objective problem. Table 3 displays the experimentation results for six test cases. The (DA) algorithm [2] produces bias solution. The average preference rank of Male proposal—DA algorithm (GS-Male) is always the best (Table 3 Column 5) compared to all other approaches, while females get worst preference match. Similarly, Female proposal—DA algorithm (GS-Female) results in female optimal SM solution (Table 3 Column 6). In both the GS algorithms, equity and social welfare score are poor as they produce biased solutions.

In few problem instances, [15] gives the same results as Gale-Shapley DA algorithm even though it solves linear program without any objective function. Our approach explicitly address the problem of multi-objectives using goal programming and finds Pareto-optimal stable solutions. Hence, our approach produces best results compared to other approaches (see Table 3) in all the instances. Also the average preference rank of men and women using our approach is fair.

**Table 3** Comparison of Our Approach with Prior Art

Test case	Method	Equity	Welfare	Avg. rank men	Avg. rank women
(1)	(2)	(3)	(4)	(5)	(6)
P50	GS-Male	371	679	4.2	9.1
	GS-Female	405	737	10.7	3.8
	Vande Vate	371	679	4.2	9.1
	Our approach	277	667	6.1	7.0
P100	GS-Male	2141	2930	7.1	14.8
	GS-Female	3684	3844	33.3	2.7
	Vande Vate	1221	2107	13.1	7.8
	Our approach	940	2083	10.1	10.0
P200	GS-Male	3276	6134	9.9	20.5
	GS-Female	6610	8324	36.7	4.6
	Vande Vate	6610	8324	36.7	4.6
	Our approach	2839	5739	17.5	11.0
P300	GS-Male	5226	10818	13.4	22.4
	GS-Female	19981	22107	69.7	3.6
	Vande Vate	4934	10222	19.1	14.7
	Our approach	4769	10177	15.4	18.3
P400	GS-Male	11234	18058	10.9	34.1
	GS-Female	29730	33258	78.3	4.5
	Vande Vate	7746	15416	20.5	17.9
	Our approach	7487	15277	19.8	18.3
P430	GS-Male	10793	19819	14.3	31.6
	GS-Female	38143	41677	92.4	4.26
	Vande Vate	18740	24978	49.6	8.2
	Our approach	8340	17848	20.2	21.2

**Computational Time** From computational time front, Gale-Shapley is very efficient  $O(n^2)$  than Vande-Vate using revised simplex to solve the linear program. In our approach, one has to solve 2 linear assignment problems and then with different weights solve the GP problem. For problem size of 430, the computation time is as follows: GS-Male (1.025 s), GS-Female (0.5 s), Vande-Vate (CPLEX run-time 117 s) and total time for GP is 300.7 s. These methods though do not address the multi-objective SM problem are quite efficient and in some cases give comparable results. Genetic Algorithm [8] took significant time i.e. 24 min for very-small size problem (20). With our GP approach, we could find Pareto-optimal SM solution in 4.2 s. Thus, our approach gives substantial reduction in the computational time compared to evolutionary algorithm.



**Fig. 1** Chart A shows Pareto-optimal solutions. Chart B shows SM solutions obtained using different approaches

**Results Validity** To check the validity of our results, we run 10 replications for a specific test case (P100). Each replication comprised of different data set which was solved using the process discussed before. We got an average improvement in Equity score as (30%) and Social Welfare score as (14.5%) compared to both Gale-Shapely and Vande-Vate approaches.

**Pareto-optimal Solutions** Fig. 1 displays the Pareto-optimal SM solutions for a particular problem instance (P100), which are obtained by varying weights in GP. Chart B in Fig. 1 displays better solutions obtained using GP compared to Gale-Shapley [2] and Vande-Vate [15].

## 5 Summary

Applications of stable matching are wide. In practice, other matching criteria such as fairness and welfare, along with stability play an important role. The local search methods addressed in the literature such as Genetic Algorithm (GA) and Neighbourhood Search (NS), were found to be quite inefficient for solving even small-size problems. The classical methods though do not explicitly address the SMMO problem provided better results compared to the GA and NS. In this paper, we proposed a Goal Programming based approach which captures the multi-objective criteria (Equity and welfare) in the SM problem. The target values in GP are obtained by solving the linear assignment models. Using this approach one can solve large size SMMO problem. Our approach in comparison with [2] and [15] produced comparable results and in many cases provided significant improvement in solution quality.

## References

1. Roth, A.E.: The evolution of the labor market for medical interns and residents: a case study in game theory. *J. Polit. Econ.* **92**(6), 991–1016 (1984)
2. Gale, D., Shapley, L.S.: College admissions and the stability of marriage. *Am. Math. Mon.* **69**(1), 9–15 (1962)
3. Irving, R.W.: An efficient algorithm for the stable roommates problem. *J. Algorithms* **6**(4), 577–595 (1985)
4. Firat, M., Hurkens, C., Laugier, A.: Stable multi-skill workforce assignments. *Ann. Oper. Res.* **213**(1), 95–114 (2014)
5. Gharote, M., Patil, R., Lodha, S., Raman, R.: Assignment of trainees to software project requirements: a SM based approach. *Comput. Ind. Eng.* **87**, 228–237 (2015)
6. Chen, N.: On computing pareto stable assignments. In: *STACS'12 (29th Symposium on Theoretical Aspects of Computer Science)*, vol. 14, pp. 384–395. LIPIcs (2012)
7. Gharote, M., Patil, R., Lodha, S.: Minimal cost stable workforce allocation in presence of ties. In: *International Conference on IEEE (IEEM)*, pp. 1146–1150 (2016)
8. Kimbrough, S.O., Kuo, A.: On heuristics for two-sided matching: revisiting the stable marriage problem as a multiobjective problem. In: *Proceedings of the 12th GECCO Conference*, pp. 1283–1290. ACM (2010)
9. Jahanshahloo, G.R., Afzalinejad, M.: Goal programming in the context of the assignment problem and a computationally effective solution method. *Appl. Math. Comput.* **200**(1), 34–40 (2008)
10. Roth, A.E., Rothblum, U.G., Vande, J.H.: Vate. Stable matchings, optimal assignments, and linear programming. *Math. Oper. Res.* **18**(4), 803–828 (1993)
11. Haas, C.: *Incentives and two-sided matching-engineering coordination mechanisms for social clouds*, vol. 12. KIT Scientific Publishing (2014)
12. Irving, R.W., Leather, P., Gusfield, D.: An efficient algorithm for the optimal stable marriage. *J. ACM (JACM)* **34**(3), 532–543 (1987)
13. Iwama, K., Miyazaki, S.: A survey of the stable marriage problem and its variants. In: *International Conference on ICKS 2008*, pp. 131–136. IEEE (2008)
14. Gelain, M., Pini, M.S., Rossi, F., Venable, K.B., Walsh, T.: Local search approaches in sm problems. *Algorithms* **6**(4), 591–617 (2013)
15. Vate, V.J.H.: Linear programming brings marital bliss. *Oper. Res. Lett.* **8**(3), 147–153 (1989)

# On Relation of Possibly Efficiency and Robust Counterparts in Interval Multiobjective Linear Programming

Milan Hladík

**Abstract** We investigate multiobjective linear programming with uncertain cost coefficients. We assume that lower and upper bounds for uncertain values are known, no other assumption on uncertain costs is needed. We focus on the so called possibly efficiency, which is defined as efficiency of at least one realization of interval coefficients. We show many favourable properties including existence, low computational performance of determining possibly efficient solutions, convexity of the dominance cone or connectedness of the efficiency set. In the second part, we discuss robust optimization approach for dealing with uncertain costs. We show that the corresponding robust counterpart is closely related to possible efficiency.

**Keywords** Interval linear programming · Multiobjective linear programming  
Robust optimization

## 1 Introduction

Many real-life problems suffer from various kind of uncertainty, including inexact measurements, vague, categorized, estimated or even partially unknown data. Decision making under uncertainty thus became a sound research area; see, e.g., a recent tutorial [18]. In this paper, we focus on multiobjective linear programming with cost coefficients to be known with interval uncertainty only.

Consider a multiobjective linear programming (MOLP) problem

$$\min_{x \in \mathcal{M}} Cx, \quad (1)$$

where  $C \in \mathbb{R}^{s \times n}$  and  $\mathcal{M} \neq \emptyset$  is a convex polyhedron. We assume that it takes the form of

---

M. Hladík (✉)  
Department of Applied Mathematics, Charles University,  
Malostranské nám. 25, 11800 Prague, Czech Republic  
e-mail: hladik@kam.mff.cuni.cz



$$\mathcal{M} := \{x \in \mathbb{R}^n \mid Ax \leq b\}, \tag{2}$$

where  $A \in \mathbb{R}^{m \times n}$  and  $b \in \mathbb{R}^m$ . Other variants are handled analogously.

Practically, there can hardly exist a feasible solution that minimizes simultaneously all objectives. That is why a common concept of a solution is the so called efficient (or Pareto optimal, or non-dominated) solution [4], which is a solution that cannot be simultaneously improved in all objectives. We will employ the relation  $u \preceq v$  with the meaning  $u \leq v$  and  $u \neq v$ .

**Definition 1** A feasible solution  $x^*$  is called *efficient* if there is no  $x \in \mathcal{M}$  such that  $Cx \preceq Cx^*$ .

An important characterization of efficient solution is that they are exactly the optimal solutions obtained by positive weighted sum scalarization of (1).

**Theorem 1** A point  $x^* \in \mathcal{M}$  is efficient if and only if it is an optimal solution of  $\min_{x \in \mathcal{M}} \lambda^T Cx$  for some  $\lambda > 0$ .

### 1.1 Interval Objectives

An interval matrix is defined as

$$C := \{C \in \mathbb{R}^{s \times n}; \underline{c}_{ij} \leq c_{ij} \leq \bar{c}_{ij} \forall i, j\},$$

where  $\underline{C}, \bar{C} \in \mathbb{R}^{s \times n}$  are given lower and upper bound matrices, respectively. We will also employ the notation of the midpoint and the radius of  $C$  defined respectively as

$$C_c := \frac{1}{2}(\underline{C} + \bar{C}), \quad C_\Delta := \frac{1}{2}(\bar{C} - \underline{C}).$$

Let  $C$  be a given interval matrix, and consider a family of MOLP problems (1) with  $C \in \mathcal{C}$ . The notion of efficiency can be adapted in two basic ways.

**Definition 2** A feasible solution  $x^*$  is called *possibly efficient* if it is efficient to (1) for at least one  $C \in \mathcal{C}$ . A feasible solution  $x^*$  is called *necessarily efficient* if it is efficient to (1) for every  $C \in \mathcal{C}$ .

Next, we denote by  $\mathcal{E}_p$  and  $\mathcal{E}_N$  the set of possibly and necessarily efficient solutions, respectively. The concepts of possibly and necessarily efficiency were pioneered by [2], and many nice properties were explored also by [12, 14, 15].

Necessary efficiency gives a guarantee that the solution remains efficient for any realization of uncertain data. However, this concept has also serious drawbacks. First, it is computationally hard even to verify whether a given feasible solution is necessarily efficient [10]. Second, a necessarily efficient solution needn't exist at all even if  $\mathcal{M}$

is nonempty and bounded. This motivated research to develop sufficient conditions for checking necessary efficiency [8, 9] and heuristics for computing necessarily efficient solutions [16].

In this paper, we focus more on possible efficiency. We show that it has many nice properties: A possible efficient solution exists under general assumptions, it can be easily computed by means of linear programming, and the possible efficiency dominance relation represents a convex polyhedral cone. We also inspect a close relation to robust counterparts of robust optimization approaches to solving interval-valued MOLP problems.

## 2 Possibly Efficiency

To find a possibly efficient solution is usually a trivial task. It is sufficient to choose any  $C \in \mathcal{C}$ , and compute any efficient solution of the corresponding realization of the MOLP problem (1). This approach can, however, fail if we choose a wrong instance having no efficient solution. The fail-safe method is described in the proof of the following theorem. It finds a possibly efficient solution or states that there is no one.

**Theorem 2** *Checking  $\mathcal{E}_p \neq \emptyset$  is a polynomial problem.*

*Proof* Let  $C \in \mathcal{C}$  and we will first be interested in the problem of finding weights  $\lambda > 0$  such that the weighted-sum scalarization  $\min_{x \in \mathcal{M}} \lambda^T Cx$  has an optimal solution (Theorem 1). The recession cone of  $\mathcal{M}$  is described by  $Ax \leq 0$ , so we need to find  $\lambda > 0$  such that

$$\min \lambda^T Cx \text{ subject to } Ax \leq 0$$

has the optimal solution in the origin. Equivalently, the system

$$-\lambda^T C = A^T v, \quad v \geq 0, \quad \lambda \geq e$$

has a solution with respect to variables  $v, \lambda$ . As  $C$  varies in  $\mathcal{C}$ , the vector  $\lambda^T C$  attains any value in the interval  $[\lambda^T \underline{C}, \lambda^T \overline{C}]$ . Thus we come to the system

$$\lambda^T \underline{C} \leq -A^T v \leq \lambda^T \overline{C}, \quad v \geq 0, \quad \lambda \geq e.$$

If it has no solution, then  $\mathcal{E}_p = \emptyset$ . If  $v, \lambda$  is any solution, then  $\min_{x \in \mathcal{M}} \lambda^T Cx$  is bounded and its optimal solution lies in  $\mathcal{E}_p$ . □

A related question is: Given  $x^* \in \mathcal{M}$ , is it possibly efficient? This question can also be answered effectively just by checking solvability of one linear system; see [12] for details.

Below, we discuss topology of the set of possibly efficient solutions.

**Theorem 3** *The set  $\mathcal{E}_p$  is a union of connected faces of  $\mathcal{M}$  provided  $\mathcal{M}$  is bounded.*

*Proof* For any  $C \in \mathcal{C}$ , the set of efficient solutions of the corresponding MOLP problem (1) is a union of connected faces of  $\mathcal{M}$ . Therefore,  $\mathcal{E}_p$  must also be formed by a union of faces of  $\mathcal{M}$ .

It remains to show connectedness. Let  $x^1, x^2 \in \mathcal{E}_p$ , that is, there are instances  $C^1, C^2 \in \mathcal{C}$ , for which they are efficient. Even more, there are  $\lambda^1, \lambda^2 > 0$  such that  $x^i$  is an optimal solution of  $\min_{x \in \mathcal{M}} (\lambda^i)^T C^i x$ ,  $i = 1, 2$ . Consider a convex combination of the objective vectors

$$c(\alpha)^T = \alpha(\lambda^1)^T C^1 + (1 - \alpha)(\lambda^2)^T C^2,$$

where  $\alpha \in [0, 1]$ . We first show that  $c(\alpha) = \lambda^T C$  for some  $\lambda > 0$  and  $C \in \mathcal{C}$  by writing

$$c(\alpha)^T = \sum_{i=1}^s \alpha \lambda_i^1 C_{i^*}^1 + (1 - \alpha) \lambda_i^2 C_{i^*}^2 \equiv \sum_{i=1}^s \lambda_i C_{i^*},$$

where  $\lambda_i := \alpha \lambda_i^1 + (1 - \alpha) \lambda_i^2 > 0$ , and

$$C_{i^*} := \frac{\alpha \lambda_i^1}{\lambda_i} C_{i^*}^1 + \frac{(1 - \alpha) \lambda_i^2}{\lambda_i} C_{i^*}^2 \in \mathcal{C}_{i^*}.$$

Now, consider the parametric LP problem

$$\min c(\alpha)^T x \text{ subject to } x \in \mathcal{M}.$$

From the theory of parametric programming [5, 6, 13] it is known, that when moving by  $\alpha$  from 0 to 1, then the optimal solutions make a connected set of faces of  $\mathcal{M}$ .  $\square$

For the real-valued MOLP problem (1),  $x^* \in \mathcal{M}$  is efficient if and only if it is not dominated by another feasible solution, that is, there is no  $x \in \mathcal{M}$  such that  $C(x - x^*) \preceq 0$ . What is an analogy of this relation in the interval case?

**Theorem 4** *Let  $x^* \in \mathcal{M}$ . Then the following are equivalent:*

- (i)  $x^* \notin \mathcal{E}_p$ ,
- (ii)  $\forall C \in \mathcal{C} \exists x \in \mathcal{M} : C(x - x^*) \preceq 0$ ,
- (iii)  $\exists x \in \mathcal{M} \forall C \in \mathcal{C} : C(x - x^*) \preceq 0$ ,
- (iv)  $\exists x \in \mathcal{M} : C_c(x - x^*) + C_\Delta |x - x^*| \preceq 0$ .

*Proof* Equivalence (i)  $\Leftrightarrow$  (ii) is by definition. Consider the interval linear system  $C(x - x^*) \preceq 0$ . Condition (ii) says that the interval system is strongly solvable, that is, each realization  $C(x - x^*) \preceq 0$  with  $C \in \mathcal{C}$  is solvable. In contrast, condition (iii)

says that the system possesses a strong solution, that is, a solution that is common to all systems  $C(x - x^*) \not\leq 0$  with  $C \in \mathcal{C}$ . Surprisingly, for standard interval linear inequalities, these two properties are equivalent (see [11, 17]), yielding condition (iv) as another characterization. Slight modification of those results generalizes to the relation “ $\not\leq$ ” as well by reformulating  $C(x - x^*) \not\leq 0$  as standard linear system  $C(x - x^*) \leq y \leq 0, e^T y = -1$ . □

This theorem shows two interesting properties. First, if  $x^* \in \mathcal{M}$  is not possibly efficient, there it is dominated by some  $x \in \mathcal{M}$  common for all realizations  $C \in \mathcal{C}$ . Second, the interval analogy of the dominance relation  $C(x - x^*) \not\leq 0$  reads  $C_c(x - x^*) + C_\Delta |x - x^*| \not\leq 0$ . Surprisingly, in spite of the absolute value in the description, the dominance cone represents a convex polyhedral cone.

**Theorem 5** *The dominance relation of possibly efficiency represents a convex polyhedral cone.*

*Proof* The dominance relation (with  $\leq$  instead of  $\not\leq$ )  $C_c x + C_\Delta |x| \leq 0$  equivalently reads

$$C_c x + C_\Delta z \leq 0, \quad -z \leq x \leq z,$$

which describes a convex polyhedral cone. □

Despite convexity and polyhedrality of the dominance cone, not all problems can be effectively solved. The dominance relation  $C_c x + C_\Delta |x| \leq 0$  can be reformulated by linear inequalities as

$$(C_c + C_\Delta \text{diag}(s))x \leq 0, \quad s \in \{\pm 1\}^n,$$

where  $\text{diag}(s)$  denotes the diagonal matrix with entries  $s_1, \dots, s_n$ . This system has exponentially many constraints. Even though some may be redundant, in general a polynomial characterization by means of linear inequalities in  $x$  needn't exist, as the following example shows.

*Example 1* Let

$$C = (1 \ [-1, 1] \ \dots \ [-1, 1]).$$

Then the dominance relation  $C_c x + C_\Delta |x| \leq 0$  equivalently reads

$$x_1 \pm x_2 \pm \dots \pm x_n \leq 0.$$

This system consists of  $2^{n-1}$  inequalities and due to symmetry no one is redundant.

### 3 Robust Counterparts

Robust solutions for multiobjective programming was investigated, e.g., in [7]. In the following sub-sections, we analyze robust counterparts for two basic multiobjective programming methods. We show that the solutions relate to possibly efficient solutions, so they needn't be robust in the traditional meaning.

#### 3.1 Robust Counterpart for the Charnes and Cooper Method

Let  $x^0 \in \mathcal{M}$  and  $C \in \mathcal{C}$ . The Charnes and Cooper method [3, 4] to compute an efficient solution (and to check efficiency of  $x^0$  in one) is based on solving the following linear program

$$\max e^T y \text{ subject to } C(x - x^0) + y \leq 0, y \geq 0, x \in \mathcal{M}. \tag{3}$$

Let  $(x^*, y^*)$  be an optimal solution. Then  $x^*$  is an efficient solution. Moreover,  $x^0$  is efficient if and only if the optimal value is 0, or equivalently,  $y^* = 0$ .

Robust approach to uncertain optimization is based on considering such feasible solutions that are resistant against all realizations of uncertain coefficients [1]. Thus, the natural robust counterpart to (3) reads

$$\max e^T y \text{ subject to } C(x - x^0) + y \leq 0 \forall C \in \mathcal{C}, y \geq 0, x \in \mathcal{M}.$$

From the results of strong solvability (solvability of each realization of interval values) of interval systems [11, 17], we rewrite the problem as

$$\max e^T y \text{ subject to } C_c(x - x^0) + C_\Delta |x - x^0| + y \leq 0, y \geq 0, x \in \mathcal{M}. \tag{4}$$

The absolute value can be reformulated by means of linear constraints, yielding a linear program

$$\max e^T y \text{ subject to } C_c(x - x^0) + C_\Delta z + y \leq 0, y \geq 0, -z \leq x - x^0 \leq z, x \in \mathcal{M}. \tag{5}$$

Relation to possibly efficiency is stated in the following observation.

**Theorem 6** *Let  $(x^*, y^*, z^*)$  be an optimal solution of (5). Then*

- (i)  $x^* \in \mathcal{E}_p$ ,
- (ii)  $x^0 \in \mathcal{E}_p$  if and only if the optimal value of (5) is 0 (i.e.,  $y^* = 0$ ).

*Proof* (i) Suppose to the contrary that there is  $x \in \mathcal{M}$  such that  $C_c(x - x^*) + C_\Delta |x - x^*| \not\leq 0$ . Then

$$\begin{aligned} & C_c(x - x^0) + C_\Delta |x - x^0| + y^* \\ & \leq C_c(x - x^*) + C_c(x^* - x^0) + C_\Delta |x - x^*| + C_\Delta |x^* - x^0| + y^* \\ & \not\leq C_c(x^* - x^0) + C_\Delta |x^* - x^0| + y^* \leq 0. \end{aligned}$$

This is a contradiction with optimality of  $y^*$ .

(ii) This is obvious in the light of Theorem 4. □

### 3.2 Robust Counterpart for the Weighted Sum Scalarization

Weighted sum scalarization is a basic computational approach in multiobjective programming. For a particular  $C \in \mathcal{C}$  and positive weights  $\lambda \in \mathbb{R}^s$ , the linear program

$$\min \lambda^T Cx \text{ subject to } x \in \mathcal{M}$$

always yields an efficient solution (Theorem 1). Let us rewrite the problem equivalently as

$$\min \alpha \text{ subject to } \alpha \geq \lambda^T Cx, x \in \mathcal{M}.$$

Now, the natural robust counterpart for uncertain values  $C \in \mathcal{C}$  draws

$$\min \alpha \text{ subject to } \alpha \geq \lambda^T Cx \forall C \in \mathcal{C}, x \in \mathcal{M}.$$

As in the above case, we have an equivalent form

$$\min \alpha \text{ subject to } \alpha \geq \lambda^T C_c x + \lambda^T C_\Delta |x|, x \in \mathcal{M},$$

or formulation as a linear program

$$\min \alpha \text{ subject to } \alpha \geq \lambda^T C_c x + \lambda^T C_\Delta z, -z \leq x \leq z, x \in \mathcal{M}. \tag{6}$$

This robust counterpart has again a strong relation to possibly efficiency.

**Theorem 7** *Let  $(x^*, z^*)$  be an optimal solution of (6). Then  $x^* \in \mathcal{E}_p$ .*

*Proof* Suppose to the contrary that there is  $x \in \mathcal{M}$  such that  $C_c(x - x^*) + C_\Delta |x - x^*| \not\leq 0$ . On account of  $\lambda > 0$ , we get

$$\lambda^T C_c(x - x^*) + \lambda^T C_\Delta |x - x^*| < 0.$$

Since  $|x - x^*| \geq |x| - |x^*|$ , we have

$$\lambda^T C_c(x - x^*) + \lambda^T C_\Delta (|x| - |x^*|) < 0,$$

from which

$$\lambda^T C_c x + \lambda^T C_\Delta |x| < \lambda^T C_c x^* + \lambda^T C_\Delta |x^*|.$$

This is a contradiction with optimality of  $x^*$ . □

## 4 Conclusion

We discussed properties of possibly efficient solutions of interval-valued MOLP. In contrast to necessarily efficient solutions, they have many convenient solutions such as guaranteed existence, low computational cost and others. They also naturally appear when considering robust counterparts of the interval optimization models.

**Acknowledgements** The author was supported by the Czech Science Foundation Grant P402/13-10660S.

## References

1. Ben-Tal, A., El Ghaoui, L., Nemirovski, A.: Robust Optimization. Princeton University Press (2009)
2. Bitran, G.R.: Linear multiple objective problems with interval coefficients. *Manage. Sci.* **26**, 694–706 (1980)
3. Charnes, A., Cooper, W.: Management Models and Industrial Applications of Linear Programming. Wiley, New York (1961)
4. Ehrgott, M.: Multicriteria Optimization, 2nd edn. Springer, Berlin (2005)
5. Gal, T.: Postoptimal Analyses, Parametric Programming, and Related Topics. McGraw-Hill, Hamburg (1979)
6. Gal, T., Greenberg, H.J. (eds.): Advances in Sensitivity Analysis and Parametric Programming. Kluwer Academic Publishers, Boston (1997)
7. Goberna, M.A., Jeyakumar, V., Li, G., Vicente-Pérez, J.: Robust solutions to multi-objective linear programs with uncertain data. *Eur. J. Oper. Res.* **242**(3), 730–743 (2015)
8. Hladík, M.: Computing the tolerances in multiobjective linear programming. *Optim. Methods Softw.* **23**(5), 731–739 (2008)
9. Hladík, M.: On necessary efficient solutions in interval multiobjective linear programming. In: Antunes, C.H., Insua, D.R., Dias, L.C. (eds.) CD-ROM Proceedings of the 25th Mini-EURO Conference Uncertainty and Robustness in Planning and Decision Making URPD 15–17 April 2010. Coimbra, Portugal, pp. 1–10 (2010)
10. Hladík, M.: Complexity of necessary efficiency in interval linear programming and multiobjective linear programming. *Optim. Lett.* **6**(5), 893–899 (2012)
11. Hladík, M.: Weak and strong solvability of interval linear systems of equations and inequalities. *Linear Algebra Appl.* **438**(11), 4156–4165 (2013)
12. Inuiguchi, M., Sakawa, M.: Possible and necessary efficiency in possibilistic multiobjective linear programming problems and possible efficiency test. *Fuzzy Sets Syst.* **78**(2), 231–241 (1996)
13. Nožička, F., Guddat, J., Hollatz, H., Bank, B.: Theorie der Linearen Parametrischen Optimierung. Akademie-Verlag, Berlin (1974)
14. Oliveira, C., Antunes, C.H.: Multiple objective linear programming models with interval coefficients—an illustrated overview. *Eur. J. Oper. Res.* **181**(3), 1434–1463 (2007)

15. Rivaz, S., Yaghoobi, M.A.: Some results in interval multiobjective linear programming for recognizing different solutions. *Opsearch* **52**(1), 75–85 (2015)
16. Rivaz, S., Yaghoobi, M.A., Hladík, M.: Using modified maximum regret for finding a necessarily efficient solution in an interval MOLP problem. *Fuzzy Optim. Decis. Mak.* **15**(3), 237–253 (2016)
17. Rohn, J.: Solvability of systems of interval linear equations and inequalities. In: Fiedler, M. et al. (eds.) *Linear Optimization Problems with Inexact Data*, chapter 2, pp. 35–77. Springer, New York (2006)
18. Wiecek, M.M., Dranichak, G.M.: Robust multiobjective optimization for decision making under uncertainty and conflict. In: Gupta, A., Capponi, A. (eds.) *Optimization Challenges in Complex, Networked and Risky Systems*, chapter 4, pp. 84–114 (2016) (INFORMS)



# Sustainable Manufacturing: An Application in the Food Industry

Maria Elena Nenni and Rosario Micillo

**Abstract** This paper aims at providing an enhancement to the decision support systems for sustainable manufacturing. We thus propose a hierarchical multi-level model to evaluate the sustainability of the production process. Our work tries to fill the gap in literature as it takes in account all the sustainability dimensions (economic, environmental and social one as well). Moreover, it is an effective decision support system as it can evaluate the impact of improvements to optimize the sustainability. We applied the model in a company operating in the food industry, by running the Analytic Hierarchy Process (AHP) method followed by a sensitivity analysis to test the model robustness.

**Keywords** Sustainable operations · Triple-bottom-line · Analytic hierarchy process (AHP)

## 1 Introduction

Managing operations in a sustainable manner has become an increasing concern and during the last years, there has been a proliferation of papers on it. One of the greatest scholar in sustainability has been Seuring, who has been constantly working on literature reviews and developing conceptual models since 2008 [14]. Other relevant contributions have focused on theory building [3], case analysis [11] and surveys [4], until to the interesting research by O'Rourke (2014) concerning how predicting and preventing unsustainable practices [10].

Otherwise previous researches are still failing in integrating the triple-bottom-line dimensions, as they still give priority to the economic dimension,

---

M.E. Nenni (✉) · R. Micillo

Department of Industrial Engineering, University of Naples Federico II, Via Claudio 21,  
80125 Naples, Italy  
e-mail: menenni@unina.it

R. Micillo

e-mail: RosarioMicillo@msn.com

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_35

345

than to the environmental one. In any case the social dimension is very often neglected [14].

Moreover, the attention is focused on the supply chain, as main source of risks of unsustainable performance. On the contrary, in our opinion it is important to realize that it is impossible to achieve a sustainable supply chain without sustainable firms and the contribution of manufacturing can be relevant.

In view of the above, we aim at exploring the relationship between business decision-making and manufacturing sustainability as our research is close to decisions for the implementation of sustainable programs at the plant level, considering the impact of environmental and social programs simultaneously with the economic one.

In this work, as first step of our research study, we aimed at individuating indicators for evaluating manufacturing sustainability. Developing metrics for sustainable manufacturing is critical to enable manufacturing companies to quantitatively measure the sustainability performance in specific manufacturing processes [8].

Several measures and metrics by means of indicators, indices, and frameworks for analysing sustainable manufacturing have also been developed. Basically, previous papers with this goal returned frameworks that are not easily operational, mostly due to the great amount of required data, or not consistent with the triple-bottom-line approach. We summarized the complete analysis of these papers in Table 1.

The model being proposed here builds on the previous research of Galal et al. in 2015 [5] but it enlarges the goal. In fact, Galal aims to maximize sustainability through the optimization of the product mix, while our model can evaluate the sustainability level of a production process in the planning period considering the resources from each operation. Successively through an Index Analysis it is possible to identify critical areas and to select the most convenient one for optimization of the outcomes.

We used Analytic Hierarchy Process (AHP) to evaluate the weight of each index of the same category. We developed then a hierarchical structure of indicators and

**Table 1** Summarization of past literature and analysis of main issues

Main issues	Papers
Not uniform measures for the three dimensions	Letmathe and Balakrishnan [8] Tsai et al. [15] Rădulescu, et al. [12]
The model can't really support the optimization of the manufacturing processes	Jayal et al. [6]
Social dimension is neglected	Li et al. [9]
Metrics are not properly interrelated	Vimal et al. [16] Joung et al. [7]
Tools for sustainability evaluation are not adequately used	Samuel et al. [13] Al-Sharrah et al. [1]

adopted an AHP approach for attaching priorities to the elements in the structure. A sensitivity analysis has been performed as well to test the model robustness.

The model has been then applied in a company of the food industry. Even if sustainability is a major issue for every industry, food and fashion ones are mainly interested because they are under constant scrutiny of the public attention and they have moreover demonstrated impacts that makes them unsustainable [2].

## 2 The Model

The index used as the objective function is composed of different hierarchal levels. At the first level, there are three pillars of sustainability, the Environmental, Economic and Social dimensions. Each of them is composed of additional indexes and sub-indexes of inferior hierarchical levels which have been further split in separate groups. The value of each index and sub-index has been determined by the resources used during the process and appropriate weight. The weight represents the relative value that stakeholders assign to an index in comparison with other indexes of the same group. We used AHP to evaluate the weight of each index of the same category. The objective function can assume a value between 0 and 1; the complete sustainability of the process would therefore assume value 1. Moreover, the sum of all indexes in each group multiplied by proper weight can assume a value between 0 and 1.

After the definition of objective function, the constraints of typical production were introduced in the model. Those constraints limit the feasibility area and represent the minimum or maximum human or material resources available for the process.

Firstly, for the application of our model we chose a product and analysed complete production process ranging from raw materials to retail market.

Secondly, we made the following assumptions: (i) Only one type of emissions is considered, which is CO<sub>2</sub>; (ii) Demand is deterministic and constant; (iii) Exact amounts of raw material required are known.

Thirdly, through the application of the triple bottom line we created the indexes (see Appendix) and we split them into different levels and groups creating a hierarchical scale. Specifically, we attributed one level to Economic and Social dimensions and two levels to Environmental dimension as shown in the Fig. 1.

Through the application of the Analytic Hierarchy Process (AHP) we could calculate the weight of the indexes, which was attributed by stakeholders. The value of the weight depends on the sample of stakeholders taken in consideration (age, social status, cultural level, etc.).

At this point we define the feasibility area by introduction of constraints that limit the objective function to be within the maximum amount of resources (see Appendix).

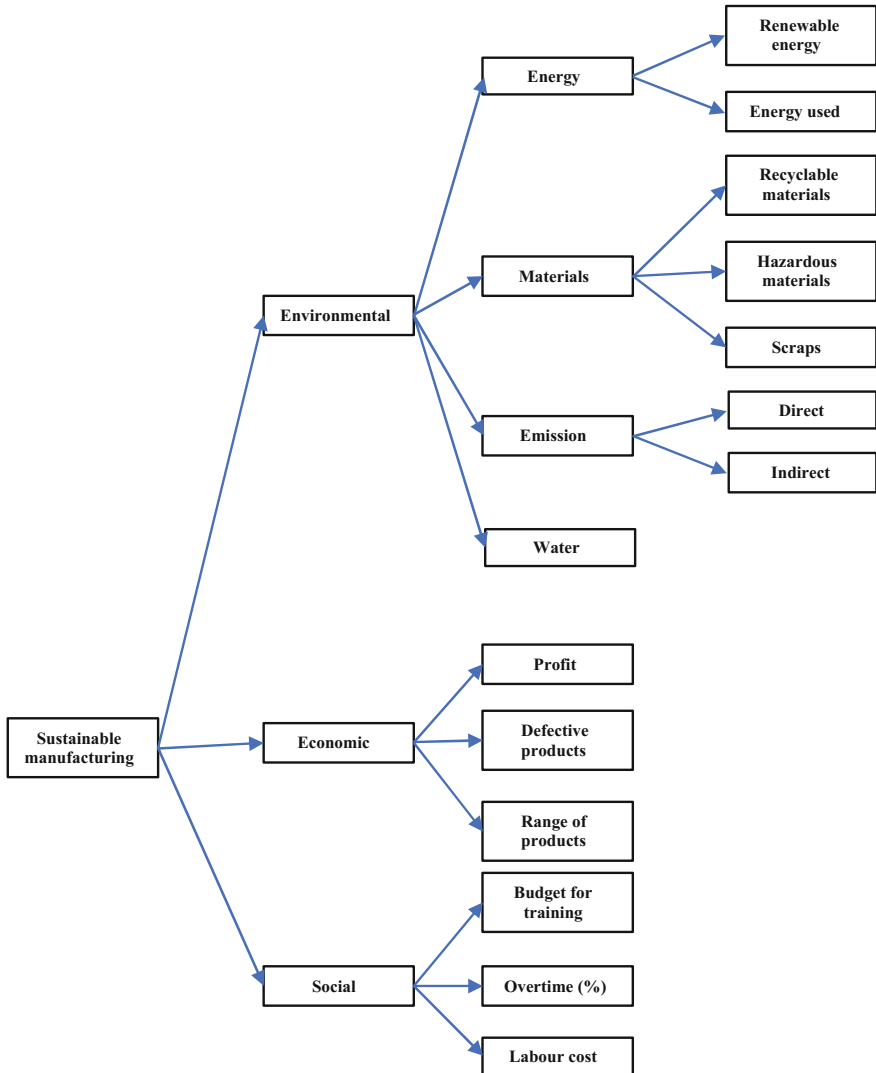


Fig. 1 Hierarchical scale used in the objective function

According to hierarchy scale, indexes and sub-indexes below the three main dimensions are aggregated as reported:

$$I_{ENV} = \sum_{i=1}^4 \sum_{j=1}^{n_i} w_{1ij} I_{1ij}$$

$$\mathbf{I}_{EC} = \sum_{l=1}^3 w_{2l} I_{2l}$$

$$\mathbf{I}_{SOC} = \sum_{l=1}^3 w_{3l} I_{3l}$$

The objective function is being expressed as:

$$SI = \frac{\sqrt{I_{ENV}^2 + I_{EC}^2 + I_{SOC}^2}}{\sqrt{\left(\sum_{i=1}^4 \sum_{j=1}^{n_i} w_{1ij}\right)^2 + \left(\sum_{l=1}^3 w_{2l}\right)^2 + \left(\sum_{l=1}^3 w_{3l}\right)^2}}$$

The outcome of this function will be the level of sustainability of the process in percentage.

In Table 2 we report the Consolidated Global Priorities, which are crucial to recognize the key areas for process sustainability maximization, and Index values multiplied by Global Consolidated Priorities. At this point, through an analysis of the results and consideration of the weights of different indexes we can introduce new data and get a forecast of the effect of possible investment on process sustainability. Obviously, keeping the resources that we are planning to invest constant, we will be reaching the maximum improvement of sustainability when targeting the index with the highest weight.

**Table 2** Consolidated global priorities and index values multiplied by global consolidated priorities

Indexes	Consolidated global priorities (GCP) (%)	Index values multiplied by GCP (%)
Renewable energy	8	1.6
Energy used	2.7	2.6
Recyclable materials	0.5	0.02
Hazardous materials	1.5	0.8
Scraps	1.5	1.5
Direct emission	2.7	2.4
Indirect emission	8	0.2
Water	3.6	2.9
Profit	36.4	31.1
Defective products	6	5.8
Range of products	14.8	5.9
Budget for training	2.3	0.01
Overtime	7.7	5.1
Labour cost	4.2	0.1

### 3 Conclusion and Future Work

Findings from our work can be relevant for assessing the operations sustainability in a plant or developing any multi-criteria decision system aiming at maximizing the sustainable performance. This has implications for decisions and processes associated with all aspects of operations management including strategy, design, planning and control, and improvement.

## Appendix

### Indices:

- i* Elements of the first hierarchy level of environmental indicators;
- j* Elements of the second hierarchy level in environmental indicators;
- k* Product type,  $k = 1, \dots, N$ ;
- l* Elements of the first hierarchy level in economic and social indicators;
- p* Input type;
- m* Hazardous material type;

### Parameters:

- MH<sub>k</sub>* Man-hours/unit weight of product *k*;
- N* Number of products;
- β<sub>k</sub>* Ratio of recyclable products;
- δ<sub>mk</sub>* Amount of hazardous material *m* in product *k* (kg);
- RT* Available regular time (h);
- W<sub>k</sub>* Amount of water consumed per unit weight of product *k* (m<sup>3</sup>/kg);
- WW<sub>k</sub>* Amount of waste water per unit weight of product *k* (m<sup>3</sup>/kg);
- Q<sub>Ck</sub>* Amount of CO<sub>2</sub> generated for producing one unit weight of product *k*;
- cp<sub>1</sub>* Emissions from 1 kWh from conventional generation (kg CO<sub>2</sub>/KWh);
- cp<sub>2</sub>* Emissions for transportation of a unit weight per unit distance (kg CO<sub>2</sub>/tKm);
- λ<sub>k</sub>* Percentage of defects of product *k*;
- D<sub>k</sub>* Demand for product *k* (kg);
- ek* Energy consumed in producing a unit weight of product *k* (KWh/kg);
- E<sub>min</sub>* Minimum allowable percentage of renewable energy used (%);
- E<sub>max</sub>** Maximum percentage of renewable energy used (%);
- B** Available working capital (Euro);
- C<sub>e</sub>** Cost of 1 kWh of electricity via renewable resources (Euro/KWh);
- C<sub>c</sub>** Price of electricity purchased from the grid (Euro/KWh);
- q<sub>pk</sub>** Quantity of input type *p* in product *k* (%);
- cp** Unit cost of input type *p* (Euro/kg);
- pk** Selling price of the unit weight of product *k* (Euro/kg);
- dk** Transportation distance of product *k* (km);

- $Q$  The maximum possible number of diversified products;
- $M$  Total manpower;
- $f_k$  Product fraction;
- $El$  Labor rate for regular time (Euro/working hours);
- $EO$  Labor rate for over time (Euro/working hours);
- $Hm$  Maximum permissible amount of hazardous material of type  $m$  (kg);
- $Ovmax$  Maximum allowed overtime expressed as a % of regular time (%);
- $Btmin$  Minimum training budget (Euro);
- $Wlij$  Weight of sub-indicator  $i$  of the  $j$ -th element of the environ. indicators
- $W2l$  Weight of the  $l$ -th element of economic indicators;
- $W3l$  Weight of the  $l$ -th element of social indicators;

**Decision Variables:**

- $Bt$  Training budget (Euro);
- $x_k$  Amount produced from product  $k$  (kg);
- $er$  Renewable energy used as the percentage of total energy necessary to produce a unit weight of product (%);
- $rk$  Amount of product  $k$  to be recycled (kg);
- $sk$  Amount of product  $k$  to be scrapped (kg);
- $Ov$  Amount of overtime needed (h);

**Indexes**

1. Renewable energy:  $I_{111} = \frac{e_r \sum_k x_k e_k}{\sum_k x_k e_k}$

2. Energy used:

$$I_{112} = 1 - \frac{\sum_k x_k e_k [C_e e_r + C_c (1 - e_r)]}{\sum_k \sum_p c_p q_{pk} x_k + \sum_k e_k x_k [C_e e_r + C_c (1 - e_r)] + RT \times M \times El + E_0 [\max(\sum_k MH_k x_k - RT \times M, 0)]}$$

3. Recyclable materials:  $I_{121} = 1 - \frac{\sum_k s_k}{\sum_k \sum_p q_{pk} x_k}$

4. Hazardous materials:  $I_{122} = 1 - \frac{\sum_k \sum_m \delta_{mk} x_k}{\sum_k \sum_p H_{m \cdot x_k}$

5. Scraps:  $I_{123} = f_k = \frac{x_k}{\sum_k x_k} \quad \forall k, x_k \neq 0$

6. Direct CO<sub>2</sub> emissions:

$$I_{131} = 1 - \frac{\sum_k Q C_k x_k}{\sum_k Q C_k x_k + c_{p1} \sum_k x_k e_k (1 - e_r) + c_{p2} \sum_k d_k x_k}$$

7. Indirect CO<sub>2</sub> emissions:

$$I_{132} = 1 - \frac{c_{p1} \sum_k x_k e_k (1 - e_r) + c_{p2} \sum_k d_k x_k}{\sum_k Q C_k x_k + c_{p1} \sum_k x_k e_k (1 - e_r) + c_{p2} \sum_k d_k x_k}$$

8. Water:  $I_{14} = 1 - \frac{\sum_k W W_k x_k}{\sum_k W_k x_k}$

## 9. Profits

$$I_{21} = \frac{\sum_k x_k p_k - \{ \sum_k \sum_p c_p q_{pk} x_k + \sum_k e_k x_k [C_e e_r + C_c (1 - e_r)] + RT \times M \times E_l + E_0 [\max(\sum_k M H_k x_k - RT \times M, 0)] + Bt \}}{\sum_k x_k p_k}$$

10. Defective products:  $I_{22} = 1 - \frac{\sum_k \lambda_k x_k}{\sum_k x_k}$

11. Range of products:  $I_{23} = \frac{\sum_{k=1}^N f_k \ln(f_k)}{\ln(1/Q)}$

## 12. Budget for Training.

$$I_{31} = \frac{Bt}{\sum_k \sum_p c_p q_{pk} x_k + \sum_k e_k x_k [C_e e_r + C_c (1 - e_r)] + RT \times M \times E_l + E_0 [\max(\sum_k M H_k x_k - RT \times M, 0)] + Bt}$$

13. Overtime:  $I_{32} = 1 - \frac{\max(\sum_k M H_k x_k - RT \times M, 0)}{Ov_{max}}$

## 14. Labour cost:

$$I_{33} = \frac{RT \times M \times E_l + E_0 [\max(\sum_k M H_k x_k - RT \times M, 0)]}{\sum_k \sum_p c_p q_{pk} x_k + \sum_k e_k x_k [C_e e_r + C_c (1 - e_r)] + RT \times M \times E_l + E_0 [\max(\sum_k M H_k x_k - RT \times M, 0)] + Bt}$$

## References

1. Al-Sharrah, G., Elkamel, A., Almansoor, A.: Sustainability indicators for decision-making and optimisation in the process industry: The case of the petrochemical industry. *Chem. Eng. Sci.* **65**(4), 1452–1461 (2010)
2. Baldwin, C. J. (ed.) *Sustainability in the food industry*. Wiley (2011)
3. Carter, C.R., Rogers, D.S.: A framework of sustainable supply chain management: moving toward new theory. *Int. J. Phys. Distrib. Logistics Manage.* **38**(5), 360–387 (2008)
4. Faruk, A.C., Lamming, R.C., Cousins, P.D., Bowen, F.E.: Analyzing, mapping, and managing environmental impacts along supply chains. *J. Ind. Ecol.* **5**(2), 13–36 (2001)



5. Galal, N.M., Moneim, A.F.A.: A mathematical programming approach to the optimal sustainable product mix for the process industry. *Sustainability* **7**(10), 13085–13103 (2015)
6. Jayal, A.D., Badurdeen, F., Dillon, O.W., Jawahir, I.S.: Sustainable manufacturing: modeling and optimization challenges at the product, process and system levels. *CIRP J. Manufact. Sci. Technol.* **2**(3), 144–152 (2010)
7. Joung, C.B., Carrell, J., Sarkar, P., Feng, S.C.: Categorization of indicators for sustainable manufacturing. *Ecol. Indic.* **24**, 148–157 (2013)
8. Letmathe, P., Balakrishnan, N.: Environmental considerations on the optimal product mix. *Eur. J. Oper. Res.* **167**(2), 398–412 (2005)
9. Li, T., Zhang, H., Yuan, C., Liu, Z., Fan, C.: A PCA-based method for construction of composite sustainability indicators. *Int. J. Life Cycle Assess.* **17**(5), 593–603 (2012)
10. O'Rourke, D.: The science of sustainable supply chains. *Science* **344**(6188), 1124–1127 (2014)
11. Pagell, M., Wu, Z.: Building a more complete theory of sustainable supply chain management using case studies of 10 exemplars. *J. Supply Chain Manag.* **45**(2), 37–56 (2009)
12. Rădulescu, M., Rădulescu, S., Rădulescu, C.Z.: Sustainable production technologies which take into account environmental constraints. *Eur. J. Oper. Res.* **193**(3), 730–740 (2009)
13. Schaltegger, S., Burritt, R.L.: Corporate sustainability accounting: a nightmare or a dream coming true? *Bus Strat. Environ.* **15**(5), 293–295 (2006)
14. Seuring, S., Müller, M.: From a literature review to a conceptual framework for sustainable supply chain management. *J. Clean. Prod.* **16**(15), 1699–1710 (2008)
15. Tsai, W.H., Chou, W.C., Hsu, W.: The sustainability balanced scorecard as a framework for selecting socially responsible investment: an effective MCDM model. *J. Oper. Res. Soc.* **60**(10), 1396–1410 (2009)
16. Vimal, K.E.K., Vinodh, S., Muralidharan, R.: An approach for evaluation of process sustainability using multi-grade fuzzy method. *Int. J. Sustain. Eng.* **8**(1), 40–54 (2015)

**Part VIII**  
**Optimization Under Uncertainty**

# The Optimal Energy Procurement Problem: A Stochastic Programming Approach

P. Beraldi, A. Violi, G. Carrozzino and M.E. Bruni

**Abstract** The paper analyzes the problem of the optimal procurement plan at a strategic level for a set of prosumers aggregated within a coalition. Electric energy needs can be covered through bilateral contracts, self-production and the pool. Signing bilateral contracts reduces the risk associated with the volatility of pool prices usually incurring higher average prices. Self-producing also reduces the risk related to pool prices. On the other hand, relying mostly on the pool might result in an unacceptable volatility of procurement cost. The problem of defining the right mix among the different sources is complicated by the high uncertainty affecting the parameters involved in the decision process (future market prices, energy demand, self-production from renewable sources). We address this more challenging problem by adopting the stochastic programming framework. The resulting model belongs to the class of two-stage model with recourse. The computational results carried out by considering a real case study shows the validity of the proposed approach.

**Keywords** Energy procurement · Stochastic programming · Risk management

## 1 Introduction

Increasing energy prices and growing concern towards environmental aspects represent driving forces pushing towards the definition of an optimized management of the electric energy resources. Nowadays, various local production/consumption units

---

P. Beraldi · A. Violi (✉) · G. Carrozzino · M.E. Bruni  
DIMEG - University of Calabria, via P. Bucci 41/C, 87036 Rende (CS), Italy  
e-mail: antonio.violi@unical.it

P. Beraldi  
e-mail: patrizia.beraldi@unical.it

G. Carrozzino  
e-mail: carrozzinogianluca@gmail.com

M.E. Bruni  
e-mail: mariaelena.bruni@unical.it

(prosumers) tend to form coalitions that can better exploit the available resources to satisfy the aggregated energy needs. The aggregator, seen as the entity responsible for managing available resources in an integrated fashion, is thus called to define the procurement plan that minimizes the overall costs. Energy demand can be supplied by three main procurement sources: bilateral contracts (with fixed prices), self-production and the day-ahead electricity market (DAEM). However, the reduction of the risk associated with the volatility of market prices usually comes at the cost of high average prices for the signed contracts. Self-producing also reduces the risk related to pool price. On the other hand, relying mostly on the market might result in an unacceptable volatility of procurement costs. Hence, the aggregator faces the problem of defining the optimal procurement by the three different sources to satisfy the energy requirement at the minimum cost. The problem has a strategic nature since the selection of the bilateral contracts along with the committed energy amount should be defined in advance with the respect to the time period in which they are actually used. Moreover, these determined values may represent input data for other problems concerning the definition of the optimal management plan of the available resources for medium and short-time horizons.

One of the main challenge in dealing with the optimal procurement problem is represented by the uncertainty. For example, the required energy demand is typically difficult to exactly predict since it refers to future needs. Market prices are known only after all producers and consumers submit their selling and bidding curves [1], and, thus, they are unknown in advance. Furthermore, the power generation from renewable resources can not be accurately predicted because it can depend, for example, on the weather conditions.

In order to appropriately address this challenging problem, we propose a stochastic programming approach (see [2] for a general introduction of SP) that provides the aggregator with a proactive procurement plan that takes simultaneously into account “all the possible circumstances” that can occur in the future. Recourse actions can be also considered to guarantee the satisfaction of the stochastic demand constraints but they are highly penalized in the objective function. Some examples of application in the energy market field are reported in [3–5]. As far as the specific procurement problem is concerned, we observe that most of the contributions that analyze the problem from both a producer’s and/or a consumer’s viewpoint are deterministic (see, for example, [6, 7], and the references therein). Carrion et al. [8] apply stochastic programming to optimality solve the electricity procurement problem faced by a large consumer under electricity price uncertainty. The authors also propose the use of the Conditional Value at Risk (CVaR) to show the trade-off between risk and expected cost, but they do not consider uncertainty affecting electricity demand. Beraldi et al. in [9] analyze the problem under electricity demand and price uncertainty. The two-stage model, referring to a short-term time horizon, is solved in a rolling-horizon fashion.

Differently from the above referred papers, here we analyze the problem from the viewpoint of an aggregator, which role has been already stated. Since the model has a general validity, we also consider the production of energy from renewable sources and the possibility to sell the energy in excess to the DAEM. Moreover, we study the

impact related to the introduction of the risk in the decision-making process, emphasizing the difference between risk-neutral and risk-averse position. The definition of this more involved stochastic programming problem along with its application to a real case study defined for an Italian aggregation may be viewed as the main contributions of the present work. The rest of the paper is organized as follows. Section 2 introduces the stochastic programming formulation for the procurement problem. Section 3 reports on the computational experiments carried out to assess the proposed approach. Concluding remarks and future research developments can be found in Sect. 4.

## 2 Problem Formulation

We consider a time horizon of a year that can be considered a “usual” planning horizon when the procurement plan for a coalition at a strategic level is defined. Month  $t$  is a single elementary period. We assume that the single hours of the different days can be classified in a set  $F$  of time blocks (e.g. peak, intermediate and off peak). We assume that the coalition energy needs can be covered by bilateral contracts (BC), self-production (from renewable and/or non renewable) units and the DAEM. Let  $N$  be the set of bilateral contracts to be evaluated. For each  $i \in N$ , we denote by  $B_{if}$  the unit price for purchasing energy from the contract  $i$  for the block  $f$  of month  $t$ . We also consider a fixed component  $FB_i$  that accounts for administrative costs. While the bilateral contract prices are known, market prices, aggregated energy needs and production from renewable sources are not known in advance. In order to explicitly address the inherent stochastic nature of the optimal procurement problem, we have adopted the stochastic programming (SP) framework. Here, the uncertain parameters are modeled as random variables defined on a given probability space  $(\Omega, \mathcal{F}, \mathcal{P})$ . Under the assumption of discrete distributions, the future uncertain evolution of DAEM prices, electricity demands and renewable production is represented by a set  $S$  of scenarios, each occurring with probability  $\pi_s$ . We denote by  $D_{tf}^s$  the uncertain overall demand and by  $P_{tf}^s$  the unitary price from purchasing energy from the market at month  $t$  and block  $f$  under scenario  $s$ . Our model also considers the possibility to sell energy in excess to the demand, so  $W_{tf}^s$  is the selling price for the market zone in which the coalition is located. We observe that demand and supply prices could be different. As far as the production from renewable sources, we denote by  $R_{tf}^s$  the production under scenario  $s$  for the time-block configuration. The scenario set provides the input data for the proposed SP formulation. In particular, we adopt a two-stage framework, where the first-stage decisions are related to procurement plan, whereas second-stage decisions model corrective actions that guarantee the fulfillment of the energy needs by drawing energy from the balance market. We denote with  $V_{tf}^{s+}$  and  $V_{tf}^{s-}$  the cost related to these actions, usually much less convenient than the DAEM prices. The following decisions formalize the procurement plan:

- $x_{if}$  the amount of electricity to purchase through contract  $i$ , time  $t$ , block  $f$ ;
- $z_i$  binary decision variable (1 if the contract is selected and 0 otherwise);
- $y_{if}, w_{if}$  amount to buy/sell from/on the DAEM for month  $t$  and block  $f$ ;
- $Q_{if}$  amount to produce at time  $t$  and block  $f$  from controllable production units;
- $\Delta_{if}^{s-}, \Delta_{if}^{s+}$  the amounts of energy required to balance (excess/shortage) the aggregated needs under scenario  $s$ .

The proposed model takes into account a set of constraints that represent real-life operating conditions.

$$\sum_{i \in N} x_{if} + y_{if} - w_{if} + Q_{if} + \Delta_{if}^{s+} - \Delta_{if}^{s-} = D_{if}^s - R_{if}^s \quad \forall t, \forall f, \forall s \quad (1)$$

$$Q_{if} \leq Q_f^{max} \quad \forall t, \forall f \quad (2)$$

$$w_{if} \leq Q_{if} + R_{if}^s \quad \forall t, \forall f, \forall s \quad (3)$$

$$LB_{if} z_i \leq x_{if} \leq UB_{if} z_i \quad \forall i, \forall t, \forall f \quad (4)$$

$$\sum_{i \in N} z_i \leq N^{max} \quad (5)$$

Constraints (1) represent the energy balance for each time block of each month and under each scenario: the overall quantity procured, by bilateral contracts, by DAEM and produced by conventional systems, minus the energy eventually sold, should satisfy the overall demand, reduced by the production from renewable plants, also considering the possibility of an adjustment through bids on secondary markets. Conditions (2) are capacity limits on the quantity that can be produced by conventional systems, while constraints (3) impose that the energy sold on the DAEM cannot be greater than the self-produced energy. The energy bought from each bilateral contract is bounded by means of (4). With constraint (5) a maximum number of active bilateral contracts is set.

The final aim of the aggregator is to minimize the total expected procurement cost, but at the same time the minimization of the risk related to a long term planning. We consider as risk the excess of costs w.r.t. a target for the entire planning horizon. The two goals can be potentially conflicting in a scenario based formulation, so we have adopted a “mean-risk” structure for the objective function:

$$\min \left[ (1 - \lambda) \sum_{s \in S} \pi_s C_s + \lambda CVaR_\beta \right], \quad (6)$$

where the first term is the expected value of the overall cost,  $CVaR_\beta$  is a risk measure and  $\lambda$  is a parameter representing the risk-aversion attitude of the decision-maker. The greater the value of  $\lambda$  the more “conservative” the solution. The overall cost under each scenario is given by the sum of different components:

$$C_s = C^{BC} + C^{Prod} + C_s^{MKT} + C_s^{Err} \quad \forall s \tag{7}$$

$$C^{BC} = \sum_{i \in N} \left[ \sum_{t \in T} \sum_{f \in F} B_{if} x_{ift} + FB_i z_i \right] \tag{8}$$

$$C^{Prod} = \sum_{i \in T} \sum_{f \in F} PC_{if} Q_{if} \tag{9}$$

$$C_s^{MKT} = \sum_{i \in T} \sum_{f \in F} (P_{if}^s y_{if} - W_{if}^s w_{if}) \quad \forall s \tag{10}$$

$$C_s^{Err} = \sum_{i \in T} \sum_{f \in F} (V_{if}^{s+} \Delta_{if}^{s+} - V_{if}^{s-} \Delta_{if}^{s-}) \quad \forall s. \tag{11}$$

$C^{BC}$  is the cost related to the bilateral contracts, and is made of both a variable and a fixed amount, as already stated, while  $C^{Prod}$  represents the cost of the energy produced by conventional production systems. These cost components are deterministic, since they do not depend on the outcomes of uncertain parameters. On the contrary,  $C_s^{MKT}$  is the net cost (or profit) for market operations under scenario  $s$  and  $C_s^{Err}$  represents the cost for the energy balance on the secondary market. As regards the risk function, we have chosen the Conditional Value at Risk for a specific confidence level  $\beta$  (usually set at 95%), a modern and widely-adopted risk measure (see [10, 11]), which allows to have an accurate measure of potential losses and can be linearized as follows:

$$CVaR_\beta = VaR_\beta + \frac{1}{1 - \beta} \sum_{s \in S} \pi_s \sigma_s \tag{12}$$

$$\sigma_s \geq L_s - VaR_\beta \quad \forall s \tag{13}$$

Here,  $L_s$  represents the possible “loss”, that is an overall cost higher than a fixed target, and  $\sigma_s$  is an auxiliary positive variable. The resulting model belongs to the class of mixed-integer multiperiod two-stage stochastic programming problems. It is worthwhile noting that the binary variables are just related to the selection of bilateral contracts, thus their number is limited for real-life decision problems.

### 3 Computational Experiments

In this section, we report on the computational experiments carried out to assess the validity of the proposed approach. The model has been implemented by using GAMS 24.7.4<sup>1</sup> as algebraic modeling system, with CPLEX 12.6.1<sup>2</sup> as solver for the mixed integer linear problems. All the test cases have been solved on a PC Intel Core I5 (2.3 GHz) with 4 GB of RAM. As testbed for the computational experience

<sup>1</sup><http://www.gams.com>.

<sup>2</sup><https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/>.

we have considered a “virtual” coalition, made up of a large electricity prosumer, University of Calabria, which has several independent departments, and one photovoltaic (PV) plant, also located in Calabria, with a rated power of about 2 MWp. A conventional production system is available as well, with a capacity of 2 MW. We have considered a planning horizon of 12 months, starting from January 2017, and 3 time blocks for each month, according to the Italian electricity market. The expected value of coalition demand, production from the renewable plant and the market prices have been calculated by analyzing the available historical series of these data. The scenario set for each instance of the problem has been generated by using a Monte Carlo technique, which, as regards electricity prices, performs a Mean Reverting Process approach (see [12]). The overall demand and production from renewable systems have been derived starting from the hourly expected demand plus increments/decrements. We have assumed that energy demand and production are independent from market prices, according to the “price-taker” assumption. This mutual independence allows to generate the whole scenario set by merging the single scenario sets through the Cartesian product. For the computational experience we have considered the availability of 10 bilateral contracts, with different characteristics, as reported in Table 1.

Moreover, each bilateral contract has a different price variability along time: just for example, BC2 energy price in F2 goes from 34.6 €/MWh in January to 47.6 €/MWh in February. Bilateral contracts differ also for lower and upper bound on the procured energy for each time block of each month. The complete data of the bilateral contracts are available in the technical report [13]. The differences between the available bilateral contracts call for the selection of a mix of them that better can satisfy the coalition needs. Moreover, we have considered as cost target the amount for satisfying the “net” overall energy needs just by purchasing on the DAEM. Preliminary computational experiments have been carried out in order to validate the decision support the model can provide. First of all, it provides managerial insights about the energy procurement process of the entire coalition. Figure 1 shows the solution obtained for a value of the risk-aversion parameter ( $\lambda$ ) equal to 0.5, in terms of procurement decisions. The optimal mix between the possible procurement opportunities, is quite variable, on both months and time blocks. This is due to the great flexibility that the proposed model provides. As regards the energy procured by bilateral contracts, only 5 contracts have been activated, as reported in Fig. 2, which shows also the consumption planning for each month in time block F1.

**Table 1** Available bilateral contracts

		BC1	BC2	BC3	BC4	BC5	BC6	BC7	BC8	BC9	BC10
Average unit price (€/MWh)	F1	42.3	42.3	42.3	41.6	41.4	42.0	40.9	40.8	40.9	40.8
	F2	43.3	43.3	43.3	43.7	42.4	41.3	41.3	40.9	40.9	41.4
	F3	35.8	35.7	35.6	35.6	35.1	34.6	34.5	34.5	34.4	34.7
Fixed cost (€)		650	675	575	375	725	650	625	500	550	850



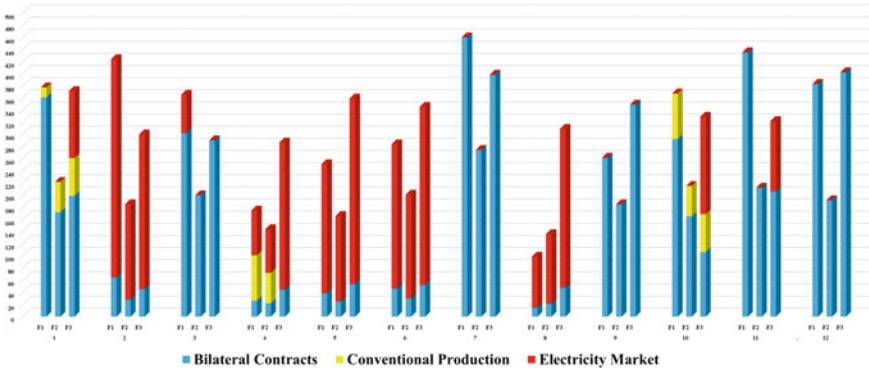


Fig. 1 Energy procurement solution for  $\lambda = 0.5$

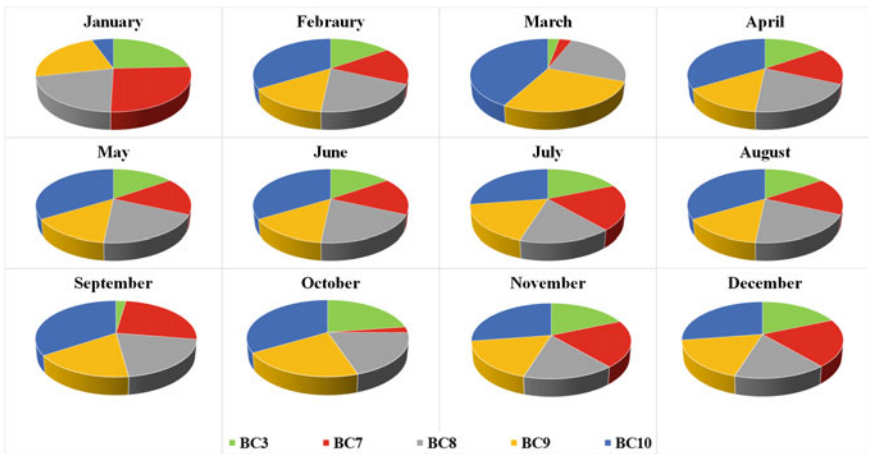


Fig. 2 Energy procured from bilateral contracts for F1

The mean-risk structure of the objective function makes the model also a useful tool to manage and control the risk exposure of the coalition in terms of cost under adverse scenarios. Figure 3 reports the efficient frontier, that is a set of “non-dominated” solutions, obtained for different values of the risk aversion parameter  $\lambda$ . We observe that a conservative attitude (high values of  $\lambda$ ) corresponds to a more expensive procurement plan, while a more risky strategy can lead to more convenient solutions, but can expose to higher losses. This result shows that our model can allow to implement different policies or to find the best trade-off between risk and economic convenience.

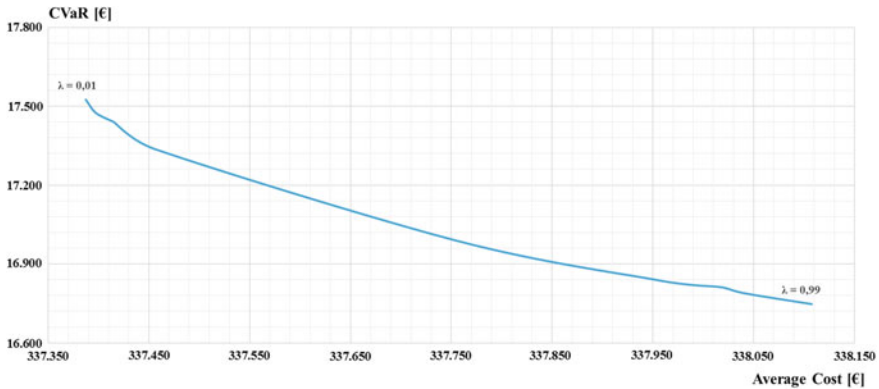


Fig. 3 Efficient frontier

## 4 Conclusions

The paper has addressed the problem of the optimal energy procurement faced by a coalition of prosumers at a strategic level. This decision plan can be viewed as the first step of a more complex and dynamic management process, including also tactical and operational phases, such as the day-by-day market bid definition. The proposed model provides the aggregator of the coalition with a decision support tool for the long-period procurement planning, in particular for the bilateral contracts selection, which usually are settled in advance for several months. Moreover, our formulation considers the uncertain nature of energy demand, market prices and production from renewable systems, and allows to control the risk related to this uncertainty. Preliminary computational experiments have shown the value of the proposed decision support and its effectiveness as a tool for the risk management.

**Acknowledgements** This work has been partially supported by Italian Minister of University and Research with the grant for research project PON03PE\_00050\_2 “DOMUS ENERGIA - Sistemi Domotici per il Servizio di Brokeraggio Energetico”.

## References

1. Beraldi, P., Conforti, D., Triki, C., Violi, A.: Constrained auction clearing in the Italian electricity market. *4OR*. **2**, 35–51 (2004)
2. Ruszczyski, A., Shapiro, A.: Stochastic programming. In: *Handbook in Operations Research and Management Science*. Elsevier Science, Amsterdam (2003)
3. Beraldi, P., Conforti, D., Violi, A.: A two-stage stochastic programming model for electric energy producers. *Comput. Oper. Res.* **35**(10), 3360–3370 (2008)
4. Musmanno, R., Scordino, N., Triki, C., Violi, A.: A multistage formulation for generation companies in a multi-auction electricity market. *IMA J. Manag. Math.* **21**(2), 165–181 (2010)

5. Corchero, C., Heredia, F.-J.: A stochastic programming model for the thermal optimal day-ahead bid problem with futures contracts. *Comput. Oper. Res.* **38**(11), 1501–1512 (2011)
6. Conejo, A., Garcia-Bertrand, R., Carrion, Caballero, A., de Andrés A.: Optimal involvement in futures markets of a power producer. *IEEE Trans. Power Syst.* **23**(2), 703–711 (2008)
7. Hatami, A.R., Seifi, H., Sheikh-El-Eslami, M.K.: Optimal selling price and energy procurement strategies for a retailer in an electricity market. *Electr. Power Syst. Res.* **79**(1), 246–254 (2009)
8. Carrion, M., Philpott, A.B., Conejo, A., Arroyo, J.M.: A stochastic programming approach to electric energy procurement for large consumers. *IEEE Trans. Power Syst.* **22**, 744–754 (2007)
9. Beraldi, P., Violi, A., Scordino, N., Sorrentino, N.: Short-term electricity procurement: a rolling horizon stochastic programming approach. *Appl. Math. Model.* **35**(8), 3980–3990 (2011)
10. Artzner, P., Delbaen, H., Eber, J.M., Heart, H.: Coherent measures of risk. *Math. Finance* **4**, 203–228 (1999)
11. Rockafellar, R., Uryasev, S.: Optimization of conditional value-at risk. *J. Risk.* **2**, 21–41 (2000)
12. Menniti, D., Scordino, N., Sorrentino, N., Violi, A.: Short-term forecasting of day-ahead electricity market price. In: 2010 7th International conference on the European Energy Market (2010)
13. Beraldi, P., Violi, A., Carrozzino, G., Bruni, M.E.: Long-term Optimal Energy Procurement: A Stochastic Programming Approach. Technical Report DIMEG (2017)

# Best and Worst Values of the Optimal Cost of the Interval Transportation Problem

R. Cerulli, C. D'Ambrosio and M. Gentili

**Abstract** We address the Interval Transportation Problem (ITP), that is, the transportation problem where supply and demand are uncertain and vary over given ranges. We are interested in determining the best and worst values of the optimal cost of the ITP among all the realizations of the uncertain parameters. In this paper, we prove some general properties of the best and worst optimum values from which the existing results derive as a special case. Additionally, we propose an Iterated Local Search algorithm to find a lower bound on the worst optimum value. Our algorithm is competitive compared to the existing approaches in terms of quality of the solution and in terms of computational time.

**Keywords** Transportation problem · Uncertainty · Interval optimization

## 1 Introduction

The Transportation Problem (TP) is a well-known optimization problem which has been studied for decades [2, 3]. The TP belongs to the general class of network flow problems [4–7] and consists of finding the minimum cost transportation plan to ship

---

R. Cerulli · C. D'Ambrosio (✉) · M. Gentili  
Department of Mathematics, University of Salerno, Giovanni Paolo II,  
132, 84084 Fisciano (SA), Italy  
e-mail: cdambrosio@unisa.it

R. Cerulli  
e-mail: raffaele@unisa.it

M. Gentili  
Industrial Engineering Department, University of Louisville,  
132 Eastern Parkway, Louisville, KY 40292, USA  
e-mail: monica.gentili@louisville.edu; mgentili@unisa.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_37

goods from a set of suppliers to a set of customers while satisfying all customer demand and not exceeding the available capacity of the suppliers. Several contributions in the literature have addressed the TP and its variants by assuming supply and demand parameters to be known with certainty. However, in the real world, the precise values for these parameters are usually not known (i.e., level of demand and supply, as well as the underlying network structure, could change over time). In this paper, we focus on the specific transportation problem where supply and demand are assumed to vary within a prespecified interval, namely the Interval Transportation Problem (ITP). Under this assumption, the total transportation cost will also vary over a range. Our focus is on determining the best and worst values of the optimal cost of ITP among all the realizations of the uncertain parameters. This problem is known in the interval literature as the Optimal Value Range Problem [8]. In particular, it has been proved that finding the best optimum value of the objective function of a general linear programming problem with interval right-hand side is a polynomially solvable problem, while finding the worst optimum value is an NP-hard problem [9]. To the best of our knowledge, there exist only four contributions in the literature which specifically address the optimal value range problem for the Interval Transportation Problem. Chanas [10] presented a transformation technique to transform the ITP into a classical TP to find the best value of the optimal cost. Liu [11] constructed two mathematical models to find the best and worst optimum values. Specifically, he provided a linear programming problem formulation whose solution provides the best optimum, and he provided a non-linear mathematical formulation to determine the worst optimum. Juman and Hoque [12] provided a heuristic solution algorithm to determine a lower bound on the worst optimum. Xie et al. [1] provided a genetic algorithm for the same problem. Additionally, Xie et al. [1] provided a necessary and sufficient condition under which finding the worst optimum becomes a polynomially solvable problem.

The contribution of this paper is twofold: (i) we prove some general properties of the worst value of the optimal cost from which the existing results in [1] derive as a special case; (ii) we propose an Iterative Local Search algorithm to find a lower bound on the worst optimum value.

The remainder of the paper is organized as follows. Section 2 introduces the formal definition of the problem and some needed notations. Section 3 describes some general properties of the problem which constitute the basis for our algorithm described in Sect. 4. Our computational results are presented in Sect. 5, and further research directions are discussed in Sect. 6.

## 2 Problem Definition

Let  $I = \{1, 2, \dots, m\}$  denote the set of  $m$  suppliers, and  $J = \{1, 2, \dots, n\}$  the set of  $n$  customers. Available supply at supplier  $i$  and required demand at customer  $j$  are uncertain quantities which vary over predefined non-negative ranges,  $[s_i, \bar{s}_i]$  and

$[d_j, \bar{d}_j]$ , respectively. Finally, let  $c_{ij} \geq 0$  denote the unit transportation cost from supplier  $i \in I$  to customer  $j \in J$ . The Interval Transportation Problem (ITP) consists of finding the minimum cost transportation plan for shipping goods from each supplier to each customer such that capacity at each supplier is not exceeded and each customer demand is satisfied. The mathematical formulation of the problem is the following:

$$(ITP) : \quad \min \sum_{i \in I, j \in J} c_{ij} x_{ij} \tag{1}$$

$$s.t. \tag{2}$$

$$\sum_{j \in J} x_{ij} \leq [s_i, \bar{s}_i] \quad \forall i \in I \tag{3}$$

$$\sum_{i \in I} x_{ij} = [d_j, \bar{d}_j] \quad \forall j \in J \tag{4}$$

$$x_{ij} \geq 0 \quad \forall j \in J, \forall i \in I \tag{5}$$

where  $x_{ij}$  is the amount of goods transported from  $i$  to  $j$ . We define the pair  $(\mathbf{s}, \mathbf{d})$  to be a *scenario* of (ITP). Specifically,  $\mathbf{s}$  is an  $m$ -dimensional vector such that  $\underline{s}_i \leq s_i \leq \bar{s}_i$  for each of its components, while  $\mathbf{d}$  is an  $n$ -dimensional vector such that  $\underline{d}_j \leq d_j \leq \bar{d}_j$  for each of its components. Given a scenario  $(\mathbf{s}, \mathbf{d})$ , we denote by  $TP(\mathbf{s}, \mathbf{d})$  the corresponding classical transportation problem, whose formulation is the following:

$$TP(\mathbf{s}, \mathbf{d}) : \quad \min \sum_{i \in I, j \in J} c_{ij} x_{ij} \tag{6}$$

$$s.t. \tag{7}$$

$$\sum_{j \in J} x_{ij} \leq s_i \quad \forall i \in I \tag{8}$$

$$\sum_{i \in I} x_{ij} = d_j \quad \forall j \in J \tag{9}$$

$$x_{ij} \geq 0 \quad \forall j \in J, \forall i \in I \tag{10}$$

We denote by  $F(\mathbf{s}, \mathbf{d})$  the feasible region of  $TP(\mathbf{s}, \mathbf{d})$  and by  $z(\mathbf{s}, \mathbf{d})$  its optimal transportation cost. Note that  $F(\mathbf{s}, \mathbf{d}) \neq \emptyset$  if and only if  $\sum_{i \in I} s_i \geq \sum_{j \in J} d_j$ . We are interested in determining the range  $[\underline{z}, \bar{z}]$  where:

$$\underline{z} = \{ \min z(\mathbf{s}, \mathbf{d}) : \forall (\mathbf{s}, \mathbf{d}) \text{ s.t. } F(\mathbf{s}, \mathbf{d}) \neq \emptyset \} \tag{11}$$

$$\bar{z} = \{ \max z(\mathbf{s}, \mathbf{d}) : \forall (\mathbf{s}, \mathbf{d}) \text{ s.t. } F(\mathbf{s}, \mathbf{d}) \neq \emptyset \} \tag{12}$$

The range  $[\underline{z}, \bar{z}]$  constitutes the best and the worst optimal costs, respectively, of ITP computed among all the feasible scenarios. Chanas [10] and Liu [11] showed that  $\underline{z}$  can be determined in polynomial time by solving an appropriate linear pro-

gramming problem. Xie et al. [1] showed that  $\bar{z}$  can be found in polynomial time under the assumption that  $\sum_{i \in I} \bar{s}_i \geq \sum_{j \in J} \bar{d}_j$ , and its value is such that  $\bar{z} = z(\underline{\mathbf{s}}, \bar{\mathbf{d}})$ . Additionally, Xie et al. [1] provided a genetic algorithm to find a lower bound of  $\bar{z}$ , that is, a value  $\hat{z} \leq \bar{z}$ .

### 3 Properties of $\bar{z}$

In this section, we prove some general properties of problem (12). These properties constitute the basis for our Iterated Local Search algorithm for finding a lower bound of  $\bar{z}$ .

The lemma below follows directly from the fact that if the sum of the upper bounds of the supplier capacity is equal to the sum of the lower bounds of the demand, then the only feasible scenario is  $(\bar{\mathbf{s}}, \underline{\mathbf{d}})$  and  $\underline{z} = \bar{z}$ :

**Lemma 1** *Given ITP such that  $\sum_{i \in I} \bar{s}_i = \sum_{j \in J} \underline{d}_j$ , then  $\bar{z} = z(\bar{\mathbf{s}}, \underline{\mathbf{d}})$ .*

Lemma 2 below states that if we consider scenarios with the same demand values, then the optimal value of the objective function increases (or remains constant) when the level of the supply decreases.

**Lemma 2** *If  $(\mathbf{s}^1, \mathbf{d})$  and  $(\mathbf{s}^2, \mathbf{d})$  are two scenarios of ITP such that  $s_i^1 \leq s_i^2, \forall i \in I$ , then  $z(\mathbf{s}^1, \mathbf{d}) \geq z(\mathbf{s}^2, \mathbf{d})$ .*

*Proof* If  $s_i^1 \leq s_i^2, \forall i \in I$  then  $F(\mathbf{s}^1, \mathbf{d}) \subseteq F(\mathbf{s}^2, \mathbf{d})$  and hence the thesis follows.  $\square$

Theorem 1 below states that if we consider scenarios with the same supply values, then the optimal value of the objective function decreases (or remains constant) when the level of the demand decreases.

**Theorem 1** *If  $(\mathbf{s}, \mathbf{d}^1)$  and  $(\mathbf{s}, \mathbf{d}^2)$  are two scenarios of ITP such that  $d_j^2 \leq d_j^1, \forall j \in J$ , then  $z(\mathbf{s}, \mathbf{d}^2) \leq z(\mathbf{s}, \mathbf{d}^1)$ .*

*Proof* Let us assume by contradiction that  $z(\mathbf{s}, \mathbf{d}^2) > z(\mathbf{s}, \mathbf{d}^1)$  and let  $x^2$  and  $x^1$  be the corresponding optimal points. By the hypothesis  $d_j^1 \geq d_j^2, \forall j \in J$ , w.l.o.g. we can assume that there exists at least one customer  $j^*$  such that  $d_{j^*}^1 > d_{j^*}^2$ . Let us consider the quantity  $\Delta_{j^*} = d_{j^*}^1 - d_{j^*}^2 > 0$ . Starting from  $x^1$ , we can build a new feasible solution  $\hat{x}$  for the scenario  $(\mathbf{s}, \mathbf{d}^2)$  as follows:

- $\hat{x}_{ij} = x_{ij}^1, \forall j \in J, j \neq j^*, \forall i \in I$
- $\hat{x}_{ij^*} = x_{ij^*}^1 - \delta_i, \forall i \in I$ , such that:  $\delta_i \geq 0, \forall i \in I, \hat{x}_{ij^*} \geq 0, \forall i \in I, \forall j \in J$  and  $\sum_{i \in I} \delta_i = \Delta_{j^*}$ .

The new solution  $\hat{x}$  is feasible for  $(\mathbf{s}, \mathbf{d}^2)$  since  $\sum_{j \in J} \hat{x}_{ij} \leq \sum_{j \in J} x_{ij}^1 \leq s_i, \forall i \in I$  and  $\sum_{i \in I} \hat{x}_{ij} = d_j^2, \forall j \in J$  by construction. Let us denote by  $\hat{z} = \sum_{i \in I, j \in J} c_{ij} \hat{x}_{ij}$ . Since the

cost coefficient  $c_{ij} \geq 0, \forall i \in I, \forall j \in J$ , it follows that  $\hat{z} < z(\mathbf{s}, \mathbf{d}^1)$ . This would imply that  $\hat{z} < z(\mathbf{s}, \mathbf{d}^2)$  which is a contradiction since  $z(\mathbf{s}, \mathbf{d}^2)$  is the minimum cost for scenario  $(\mathbf{s}, \mathbf{d}^2)$ .  $\square$

From Lemma 2 and Theorem 1, the theorem proved in [1] results as a corollary:

**Corollary 1** *Given ITP such that  $\sum_{i \in I} \underline{s}_i \geq \sum_{j \in J} \bar{d}_j$  then  $\bar{z} = z(\underline{\mathbf{s}}, \bar{\mathbf{d}})$ .*

## 4 Our Iterated Local Search Algorithm

Our solution algorithm is a local search based metaheuristic which starts from a feasible solution and iteratively tries to improve it by an intelligent exploration of the solution space. For this reason, a neighborhood structure is required which is used to explore the search space. We define the following neighborhood structures:

**$k$ - Plus -Neighborhood:**  $N_k^+(\mathbf{s}, \mathbf{d})$

*A scenario  $(\hat{\mathbf{s}}, \hat{\mathbf{d}}) \in N_k^+(\mathbf{s}, \mathbf{d})$  if and only if there exists  $i^* \in I$  and  $j^* \in J$  such that: (1)  $\hat{s}_i = s_i, \forall i \neq i^*$ , (2)  $\hat{d}_j = d_j, \forall j \neq j^*$ , and (3)  $\hat{s}_{i^*} = s_{i^*} + k, \hat{d}_{j^*} = d_{j^*} + k$ .*

**$k$ - Minus -Neighborhood:**  $N_k^-(\mathbf{s}, \mathbf{d})$

*A scenario  $(\hat{\mathbf{s}}, \hat{\mathbf{d}}) \in N_k^-(\mathbf{s}, \mathbf{d})$  if and only if there exists  $i^* \in I$  and  $j^* \in J$  such that: (1)  $\hat{s}_i = s_i, \forall i \neq i^*$ , (2)  $\hat{d}_j = d_j, \forall j \neq j^*$ , and (3)  $\hat{s}_{i^*} = s_{i^*} - k, \hat{d}_{j^*} = d_{j^*} - k$ .*

That is, a plus-neighbor (minus-neighbor) of a scenario  $(\mathbf{s}, \mathbf{d})$  is obtained by choosing one supplier and increasing (decreasing) its supply by a given quantity  $k$ , and by choosing a customer and increasing (decreasing) its demand by the same amount  $k$ .

Our algorithm consists of three phases: *initialization*, *intensification*, and *diversification*.

The initialization phase constructs a feasible initial solution by solving a transportation problem corresponding to a feasible scenario  $(\mathbf{s}, \mathbf{d})$  chosen according to the following method. The demands  $d_j$ , for each customer  $j \in J$ , are chosen such that  $\sum_{j \in J} d_j$  is as low as possible, and the values  $s_i$  for each supplier  $i \in I$ , are set such that  $\sum_{i \in I} s_i = \sum_{j \in J} d_j$  by solving an appropriate linear programming problem.

During the intensification phase our procedure exploits the two neighborhoods  $N_k^+(\mathbf{s}, \mathbf{d})$  and  $N_k^-(\mathbf{s}, \mathbf{d})$  by alternatively setting  $k = 1$  and  $k = 2$ . More precisely, we start the search procedure with the plus neighborhood with  $k = 1$ . When the search ends at its best local optimum, the minus neighborhood with  $k = 1$  is explored. When the search ends at its best local optimum, the plus neighborhood with  $k = 2$ , and successively the minus neighborhood with  $k = 2$ , are explored. This process is repeated until no improvement is possible for a predefined number of iterations. If this is the case, the diversification phase starts.

The diversification phase is applied whenever the search is trapped at a local maximum. During this phase, we first change the neighborhoods to be explored by varying  $k$  between 1 and a maximum number of iterations (*exploration* step). That is, the



plus and minus neighborhoods are alternately explored by increasing the value of  $k$  by one unit each time up to a maximum value which we set equal to 20 after a tuning phase. If during this exploration step, a solution better than the incumbent local maximum is found, then an intensification phase is started. Otherwise, if the value of  $k$  is increased up to 20 without any improvement, the current local maximum is shaken (*shaking* step), and an intensification phase starts on the new shaken solution. The shaking step generates a new feasible solution by solving a transportation problem corresponding to a shaken scenario obtained as follows. Let us denote by  $\hat{d}_j, \forall j \in J$ , the demands corresponding to the current local maximum solution, and let  $\Delta$  be a randomly generated positive number. The shaking step generates new demands  $d_j$  such that  $\sum_{j \in J} d_j = \sum_{j \in J} \hat{d}_j + \Delta$  and sets the values of the supplies  $s_i, \forall i \in I$  such that  $\sum_{i \in I} s_i = \sum_{j \in J} d_j$  by solving an appropriate linear programming problem. Note that the random number  $\Delta$  is generated so that the new shaken scenario is feasible.

The algorithm terminates when a predefined number of iterations without improvement is reached.

## 5 Computational Results

We compared our algorithm with the genetic algorithm provided by Xie et al. [1] on the set of 60 instances they used in their paper. The instances include problems of different sizes, namely  $2 \times 3$ ,  $3 \times 5$ ,  $4 \times 6$ ,  $5 \times 10$ ,  $10 \times 10$ , and  $20 \times 20$ . Our algorithm is coded in C++ and runs on a PC with Intel Core i5 2.4 GHz and 16.00 GB of RAM.

Table 1 compares the values returned by the two algorithms on each of the 60 instances, while Table 2 shows the normalized [13] average computational time (in seconds) for the two algorithms on each set of 10 instances of the same size. These results show that our algorithm is competitive compared to the genetic algorithm both in terms of quality of the solution and computational time.

Our iterative local search algorithm has a lower average computational time on all the sets of instances with the exception of those of size  $10 \times 10$  with an increase in the average computational time of 35%. On all the other sets of instances the improvement in the computational times achieved by our algorithm varies between 1% and 41%. The two algorithms find the same solution for 46 of 60 instances, probably because these are the optimum  $\bar{z}$  values. Our algorithm finds a better lower bound for 10 instances (instances number 1, 2, and 7 of size  $5 \times 10$ ; instances 4 and 7 of size  $10 \times 10$ ; and instances 2–6 of size  $20 \times 20$ ), while it returns a lower value than the one returned by the genetic algorithm of Xie et al. for 4 instances (instance 8 of size  $3 \times 5$ , instance 9 of size  $10 \times 10$ , and instances 7 and 10 of size  $20 \times 20$ ).

**Table 1** Comparison of the results provided by the genetic algorithm of Xie et al. [1] and our Iterated Local Search (ILS)

Instances	Xie et al.	ILS	Instances	Xie et al.	ILS	Instances	Xie et al.	ILS
<i>data2x3_1</i>	22800	22800	<i>data4x6_1</i>	27125	27125	<i>data_10x10_1</i>	36180	36180
<i>data2x3_2</i>	27390	27390	<i>data4x6_2</i>	20635	20635	<i>data_10x10_2</i>	43260	43260
<i>data2x3_3</i>	27390	27390	<i>data4x6_3</i>	23615	23615	<i>data_10x10_3</i>	38915	38915
<i>data2x3_4</i>	27210	27210	<i>data4x6_4</i>	22375	22375	<i>data_10x10_4</i>	<b>38845</b>	<b>38905</b>
<i>data2x3_5</i>	18570	18570	<i>data4x6_5</i>	19500	19500	<i>data_10x10_5</i>	50150	50150
<i>data2x3_6</i>	30900	30900	<i>data4x6_6</i>	11380	11380	<i>data_10x10_6</i>	29885	29885
<i>data2x3_7</i>	22020	22020	<i>data4x6_7</i>	17245	17245	<i>data_10x10_7</i>	<b>44100</b>	<b>44145</b>
<i>data2x3_8</i>	18450	18450	<i>data4x6_8</i>	24180	24180	<i>data_10x10_8</i>	41950	41950
<i>data2x3_9</i>	21450	21450	<i>data4x6_9</i>	24060	24060	<i>data_10x10_9</i>	<u>37465</u>	<u>37180</u>
<i>data2x3_10</i>	14130	14130	<i>data4x6_10</i>	22825	22825	<i>data_10x10_10</i>	48920	48920
<i>data3x5_1</i>	16410	16410	<i>data_5x10_1</i>	<b>25760</b>	<b>25810</b>	<i>data_20x20_1</i>	9405	9405
<i>data3x5_2</i>	14820	14820	<i>data_5x10_2</i>	<b>24980</b>	<b>25055</b>	<i>data_20x20_2</i>	<b>9015</b>	<b>9140</b>
<i>data3x5_3</i>	20650	20650	<i>data_5x10_3</i>	19635	19635	<i>data_20x20_3</i>	<b>9335</b>	<b>9405</b>
<i>data3x5_4</i>	12940	12940	<i>data_5x10_4</i>	30260	30260	<i>data_20x20_4</i>	<b>8930</b>	<b>9130</b>
<i>data3x5_5</i>	16650	16650	<i>data_5x10_5</i>	23590	23590	<i>data_20x20_5</i>	<b>9275</b>	<b>9420</b>
<i>data3x5_6</i>	16540	16540	<i>data_5x10_6</i>	23075	23075	<i>data_20x20_6</i>	<b>10220</b>	<b>10320</b>
<i>data3x5_7</i>	10195	10195	<i>data_5x10_7</i>	<b>22375</b>	<b>22740</b>	<i>data_20x20_7</i>	<u>8685</u>	<u>8630</u>
<i>data3x5_8</i>	<u>13360</u>	<u>12620</u>	<i>data_5x10_8</i>	30000	30000	<i>data_20x20_8</i>	9260	9260
<i>data3x5_9</i>	11010	11010	<i>data_5x10_9</i>	24675	24675	<i>data_20x20_9</i>	9885	9885
<i>data3x5_10</i>	12915	12915	<i>data_5x10_10</i>	39985	39985	<i>data_20x20_10</i>	<u>9225</u>	<u>9220</u>

**Table 2** Average normalized computational times (in seconds) of each algorithm on each set of 10 instances of the same size

Instance size	Xie et al.	ILS
2 × 3	1.54	1.33
3 × 5	7.70	4.51
4 × 6	12.33	9.96
5 × 10	49.31	29.62
10 × 10	92.45	124.60
20 × 20	1458.40	1439.40

## 6 Conclusions and Further Research

We addressed the problem of finding the range of the optimal cost of a transportation problem when supply and demand vary over an interval. We consider the specific version of a transportation problem with supply inequality constraints and demand equality constraints. We investigated some theoretical properties of the problem of finding the worst optimum value and presented an Iterated Local Search algorithm. Our results show that our algorithm is competitive compared to the existing approaches both in terms of quality of the returned solution and in the computational time.

We are now focusing on improving the computational performance of our Iterated Local Search algorithm and on extending the experimental analysis. We are also investigating theoretical properties and efficient algorithms for finding the range of the optimal cost for different versions of the transportation problem with interval supply and demand. In particular, we are investigating the version with supply and demand equality constraints and the version with supply equality constraints and demand inequality constraints.

## References

1. Xie, F., Butt, M.M., Li, Z., Zhu, L.: An upper bound on the minimal total cost of the transportation problem with varying demands and supplies. *Omega* **68**, 105–118 (2017)
2. Charnes, A., Cooper, W.W.: The stepping stone method of explaining linear programming calculations in transportation problems. *Manag. Sci.* **1**(1), 49–69 (1954)
3. Frank, L.: Hitchcock. The distribution of a product from several sources to numerous localities. *Stud. Appl. Math.* **20**(1–4), 224–230 (1941)
4. Bianco, L., Cerrone, C., Cerulli, R., Gentili, M.: Locating sensors to observe network arc flows: Exact and heuristic approaches. *Comput. Oper. Res.* **46**, 12–22 (2014)
5. Cerrone, C., Cerulli, R., Gentili, M.: Vehicle-id sensor location for route flow recognition: models and algorithms. *Eur. Oper. Res.* **247**(2), 618–629 (2015)
6. Carrabs, F., Cerrone, C., Cerulli, R., Gaudioso, M.: A novel discretization scheme for the close enough traveling salesman problem. *Comput. Oper. Res.* **78**, 163–171 (2017)
7. Carrabs, F., Cerrone, C., Cerulli, R., D'Ambrosio, C.: Improved Upper and Lower Bounds for the Close Enough Traveling Salesman Problem, pp. 165–177. Springer International Publishing, Cham (2017)
8. Hladik, M.: Interval linear programming: a survey. In: *Linear programming-new frontiers in theory and applications*, pp. 85–120 (2012)
9. Gabrel, V., Murat, C., Remli, N.: Linear programming with interval right hand sides. *Int. Trans. Oper. Res.* **17**(3), 397–408 (2010)
10. Chanas, S., Delgado, M., Verdegay, J.L., Vila, M.A.: Interval and fuzzy extensions of classical transportation problems. *Transp. Plan. Technol.* **17**(2), 203–218 (1993)
11. Liu, S.-T.: The total cost bounds of the transportation problem with varying demand and supply. *Omega* **31**(4), 247–251 (2003)
12. Juman, Z.A.M.S., Hoque, M.A.: A heuristic solution technique to attain the minimal total cost bounds of transporting a homogeneous product with varying demands and supplies. *Eur. J. Oper. Res.* **239**(1), 146–156 (2014)
13. [https://setiathome.berkeley.edu/cpu\\_list.php](https://setiathome.berkeley.edu/cpu_list.php)

# A Queueing Networks-Based Model for Supply Systems

Massimo De Falco, Nicola Mastrandrea and Luigi Rarità

**Abstract** In this paper, a stochastic approach, based on queueing networks, is analyzed in order to model a supply system, whose nodes are working stations. Unfinished goods and control electrical signals arrive, following Poisson processes, at the nodes. When the working processes at nodes end, according to fixed probabilities, goods can leave the network or move to other nodes as either parts to process or control signals. On the other hand, control signals are activated during a random exponentially distributed time and they act on unfinished parts: precisely, with assigned probabilities, control impulses can move goods between nodes, or destroy them. For the just described queueing network, the stationary state probabilities are found in product form. A numerical algorithm allows to study the steady state probabilities, the mean number of unfinished goods and the stability of the whole network.

**Keywords** Queueing networks · Supply systems · Product–form solution

## 1 Introduction

A great interest has always been devoted to model industrial processes managed by supply systems. Such an exigence has become higher due to the necessity of obtaining safe and fast processes that could avoid, in some way, unwished situations, such as bottlenecks, dead times, and so on.

---

M. De Falco · N. Mastrandrea  
Dipartimento di Scienze Aziendali - Management & Innovation Systems,  
University of Salerno, Via Giovanni Paolo II, 132, 84084 Fisciano (SA), Italy  
e-mail: mdefalco@unisa.it

N. Mastrandrea  
e-mail: nmastrandrea@unisa.it

L. Rarità (✉)  
CO.RI.SA, COnsorzio RIcerca Sistemi Ad Agenti, University of Salerno,  
Via Giovanni Paolo II, 132, 84084 Fisciano (SA), Italy  
e-mail: lrarita@unisa.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_38

Various mathematical models have been studied for this aim. Several approaches are continuous and mainly based on differential equations, see for instance [3, 9]. In these cases, supply chains are characterized by: parts dynamics described by conservation laws; Queues, that are in front of each supplier and are defined by ordinary differential equations. There are also other models, that focus on individual parts and deal with exponential queueing networks. A theoretical example is given in [8] for waiting lines, while various applications for manufacturing systems are in [4, 11], with emphasis on possible numerical approaches in [10]. In this direction, some variants have been studied, such as the “G-networks” (see [5, 6]), introduced by Gelenbe, motivated by analogies with neural networks, and interested by the simultaneous presence of positive customers, negative customers, signals and triggers. Positive customers are the common ones, who join a queue for a service, and they can be destroyed by a negative customer arriving at a non-empty queue. Triggers displace positive customers from a queue to another one, while a signal behaves either as a negative customer or a trigger. Exhaustive descriptions of G-networks are in [1, 2], where exact solutions are found in product form.

In this paper, focusing on some G-networks described in [7], we consider a queueing network, that models a supply system, characterized by either parts dynamics or control electrical signals in the working stations. Unfinished goods and control impulses, these last ones generated by a Control Station (CS), arrive, following two different and independent Poisson processes, at each node from outside the network. Parts are processed one by one at each node, and service times of the unfinished goods are exponentially distributed. After the working process, a part travels from a node to another one with fixed probabilities as either a good to process or a control signal, or leaves the network. The activation time of a control electrical signal is exponentially distributed. Activated impulses with fixed probabilities either move a good from the node they are activated to another one or destroy an unfinished good. For such a queueing network, the stationary state distribution is obtained in product form, and numerical results, also focusing on the ergodicity condition, are then computed. Notice that the queueing network analysed in this paper is a model of a system for car engines inside a real Italian company. The main advantage, with respect to the existing models in the scientific literature, is the extension of the G-networks by introducing phenomena that usually occur in the real industrial systems.

The paper is structured as follows. Section 2 describes the supply system and its mathematical formulation. Section 3 contains some numerical results. Conclusions end the paper in Sect. 4.

## 2 A Model for Supply Systems

Focus on a supply system, modelled via a queueing network with the following features:

- Each node represents a working station, that receives raw material flows of either external type (flows from outside the network) or internal one (flows from nodes inside the network). Materials are processed one by one at each node, characterized by an own working frequency and an infinite buffer.
- A Control Station (CS) provides each node some electrical control impulses that rule dynamics for the working stations.
- Beside the control signals generated by the CS, each node has a set of non-active impulses, that have their own frequency action and are activated only if necessary. In particular, if a node has not goods to process (namely, it is empty), the activation of a control impulse has no effects, the impulse is disabled and is not activated anymore.
- An unfinished good, once it has been processed in a node  $i$ , either leaves the network or moves to another node  $j$ . Inside node  $j$ , the good can be further manufactured, or behave like a control signal. In this last case the unfinished part can destroy a good, which is inside node  $j$ , or move the good itself to another node  $k$ .

From a mathematical point of view, the queueing network is identified by the couple  $(\mathcal{V}, \mathcal{E})$ , where  $\mathcal{V} = \{0, 1, 2, \dots, M\}$  and  $\mathcal{E} = \bigcup_{i \in \mathcal{V}, j \in \mathcal{V}} \{\varepsilon_{ij}\}$  represent, respectively, the set of nodes and arcs. Precisely, node 0 indicates the external of the network, while node  $i, i = 1, \dots, M$ , is a generic working station, which belongs to the queueing network;  $\varepsilon_{ij}$  is the arc that connects nodes  $i$  and  $j, i \in \mathcal{V}, j \in \mathcal{V}$ .

The queueing network has  $M$  working stations with infinite buffers. External arrival flows are independent Poisson processes. Precisely, the arrival rates of external unfinished goods and electrical control impulses, generated by the CS, at node  $i, i = 1, \dots, M$ , are, respectively, indicated by  $A_{0i}^1$  and  $A_{0i}^2$ . Each node  $i$  has one server, hence goods are processed one by one with a frequency  $b_i^1$ . An unfinished good, that leaves node  $i$ , moves to node  $j, j = 1, \dots, M$ , with: probability  $\gamma_{ij}^1$  as a part that has to be processed inside node  $j$ ; probability  $\gamma_{ij}^2$  as a control impulse for node  $j$ . Finally, the unfinished part leaves the network with probability  $\gamma_{i0} = 1 - \sum_{j=1}^M (\gamma_{ij}^1 + \gamma_{ij}^2)$ . Define the matrices  $\mathbf{G}^1 := (\gamma_{ij}^1)$  and  $\mathbf{G}^2 := (\gamma_{ij}^2)$ . Then,  $\mathbf{G} = \mathbf{G}^1 + \mathbf{G}^2 := (\gamma_{ij}^1 + \gamma_{ij}^2)$  represents the transition matrix of a Markov chain for the dynamics of parts.

A control impulse is activated at a random instant of time  $t$ . An impulse, sent to node  $i$ , works in  $]t, t + \Delta[$  with probability  $b_i^2(k) \Delta + o(\Delta)$ , provided that  $k$  non-activated signals are inside node  $i$  at the time instant  $t$ . When the activation period ends, a control impulse: with probability  $\xi_{ij}^1$  moves a good, that is inside node  $i$ , to node  $j$  with the aim of continuing the working process; with probability  $\xi_{ij}^2$  moves to node  $j$  an unfinished part, that belongs to node  $i$ , and the moved good behaves like a control impulse inside node  $j$ . Finally,  $\xi_{i0} = 1 - \sum_{j=1}^M (\xi_{ij}^1 + \xi_{ij}^2)$  is the probability that a control impulse destroys an unfinished part inside node  $i$ . When this event occurs, the control impulse ends its own action and is not activated inside node  $i$

any more. Consider now the matrices  $\mathbf{H}^1 := \left(\xi_{ij}^1\right)$  and  $\mathbf{H}^2 := \left(\xi_{ij}^2\right)$ . Then, the matrix  $\mathbf{H} = \mathbf{H}^1 + \mathbf{H}^2 := \left(\xi_{ij}^1 + \xi_{ij}^2\right)$  is the transition matrix of a Markov chain, that focuses on all possible dynamics for control impulses.

### 2.1 Equilibrium Equations and Stationary Probabilities

The just described system is modelled by a queueing network, represented by a homogeneous Markov process  $\{Z(t), t \geq 0\}$ , with state space:

$$\zeta = \left\{ \left( (x_1, y_1), (x_2, y_2), \dots, (x_M, y_M) \right), x_i \geq 0, y_i \geq 0, i = 1, \dots, M \right\}.$$

Notice that the state  $\left( (x_1, y_1), (x_2, y_2), \dots, (x_M, y_M) \right)$  indicates that, for a defined instant of time, node  $i, i = 1, \dots, M$ , has  $x_i$  unfinished goods and  $y_i$  non-activated impulses. Define the quantities:

$$\mathbf{x} := (x_1, x_2, \dots, x_M), \quad \mathbf{y} := (y_1, y_2, \dots, y_M),$$

$$(\mathbf{x}, \mathbf{y}) := \left( (x_1, y_1), (x_2, y_2), \dots, (x_M, y_M) \right), \quad A_0^1 := \sum_{i=1}^M A_{0i}^1, \quad A_0^2 := \sum_{i=1}^M A_{0i}^2,$$

and indicate by  $\mathbf{e}_i$  the vector, whose  $i$ -th component is 1 while the other ones are zero. Assume that  $\pi(\mathbf{x}, \mathbf{y})$  is the stationary probability of the state  $(\mathbf{x}, \mathbf{y})$ , namely the probability that the queueing network has, for large times,  $x_i$  unfinished goods and  $y_i$  non-activated impulses inside node  $i, \forall i = 1, \dots, M$ . If  $\pi(\mathbf{x}, \mathbf{y})$  exists, then the following Chapman-Kolmogorov equations system holds:

$$\begin{aligned} \pi(\mathbf{x}, \mathbf{y}) & \left( A_0^1 + A_0^2 + \sum_{i=1}^M b_i^1 (1 - \gamma_{ii}^1) u(x_i) + \sum_{i=1}^M b_i^2 (y_i) \right) = \sum_{i=1}^M \pi(\mathbf{x} - \mathbf{e}_i, \mathbf{y}) A_{0i}^1 u(x_i) + \\ & + \sum_{i=1}^M \pi(\mathbf{x}, \mathbf{y} - \mathbf{e}_i) A_{0i}^2 u(y_i) + \sum_{i=1}^M \pi(\mathbf{x} + \mathbf{e}_i, \mathbf{y}) b_i^1 \gamma_{i0}^1 u(x_i + 1) + \\ & + \sum_{i=1}^M \pi(\mathbf{x} + \mathbf{e}_i, \mathbf{y} + \mathbf{e}_i) b_i^2 (y_i + 1) \xi_{i0} + \sum_{i=1}^M \pi(\mathbf{x}, \mathbf{y} + \mathbf{e}_i) b_i^2 (y_i + 1) (1 - u(x_i)) + \\ & + \sum_{i=1}^M \sum_{j=1, j \neq i}^M \pi(\mathbf{x} + \mathbf{e}_i - \mathbf{e}_j, \mathbf{y}) b_i^1 \gamma_{ij}^1 u(x_i + 1) u(x_j) + \end{aligned}$$

$$\begin{aligned}
 & + \sum_{i=1}^M \sum_{j=1}^M \pi(\mathbf{x} + \mathbf{e}_i, \mathbf{y} - \mathbf{e}_j) b_i^1 \gamma_{ij}^2 u(x_i + 1) u(y_j) + \\
 & + \sum_{i=1}^M \sum_{j=1}^M \pi(\mathbf{x} + \mathbf{e}_i - \mathbf{e}_j, \mathbf{y} + \mathbf{e}_i) b_i^2 (y_i + 1) \xi_{ij}^1 u(x_j) + \\
 & + \sum_{i=1}^M \sum_{j=1, j \neq i}^M \pi(\mathbf{x} + \mathbf{e}_i, \mathbf{y} + \mathbf{e}_i - \mathbf{e}_j) b_i^2 (y_i + 1) \xi_{ij}^2 u(y_j) + \\
 & + \sum_{i=1}^M \pi(\mathbf{x} + \mathbf{e}_i, \mathbf{y}) b_i^2 (y_i) \xi_{ii}^2, \quad (\mathbf{x}, \mathbf{y}) \in \zeta, \tag{1}
 \end{aligned}$$

where  $b_i^2(0) = 0$  and  $u(x)$  is a unit Heavyside function. The system (1), that allows to get the steady state probability  $\pi(\mathbf{x}, \mathbf{y})$ , has been obtained considering all transitions from/to the state  $(\mathbf{x}, \mathbf{y})$  by balancing incoming and outgoing flows for such state (similar examples are in [7]).

Now, a general product-form solution for the equations system (1) is provided. Indicate by  $A_i^1$  and  $A_i^2$  the total steady state rates of arrival of goods and control electrical impulses, respectively, at node  $i$ , and define the quantities:

$$x_i^2 := A_i^2 + b_i^1, \quad \rho_i := \frac{A_i^1}{x_i^2}, \quad q_i^2(j) := \frac{A_i^2}{b_i^2(j)}, \quad i = 1, \dots, M, j = 1, \dots, M;$$

*Remark 1* Notice that  $\rho_i$  is the stationary probability that the queue of the working station  $i$  is busy.

The following traffic equations hold (see [1, 5–7] for further details):

$$A_i^1 = A_{0i}^1 + \sum_{j=1}^M \rho_j \left( b_j^1 \gamma_{ji}^1 + A_j^2 \xi_{ji}^1 \right), \quad i = 1, \dots, M, \tag{2}$$

$$A_i^2 = A_{0i}^2 + \sum_{j=1}^M \rho_j \left( b_j^1 \gamma_{ji}^2 + A_j^2 \xi_{ji}^2 \right), \quad i = 1, \dots, M. \tag{3}$$

We get the following theorems (for ideas of the proofs, see [7]):

**Theorem 1** (Solution of traffic equations) *If matrices  $\mathbf{G}$  and  $\mathbf{H}$  are irriducible, the solution  $\{A_i^1, A_i^2\}$ ,  $i = 1, \dots, M$ , to equations (2) and (3) is unique.*

**Theorem 2** (Product-form solution for stationary probabilities) *If matrices  $\mathbf{G}$  and  $\mathbf{H}$  are irreducible and the following conditions hold:*



$$\rho_i < 1, \quad \delta_i = \sum_{y_i=0}^{+\infty} \prod_{j=1}^{y_i} q_i^2(j) < \infty, \quad i = 1, \dots, M,$$

then the Markov process  $\{Z(t), t \geq 0\}$  is ergodic and its stationary distribution is represented in product form as:

$$\pi(\mathbf{x}, \mathbf{y}) = \prod_{i=1}^M \pi_i(x_i, y_i),$$

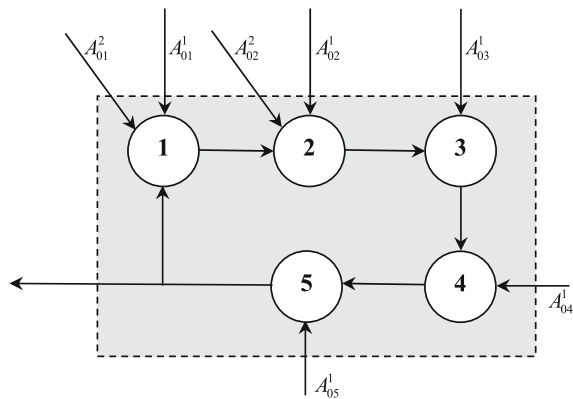
$$\pi_i(x_i, y_i) = \frac{(1 - \rho_i) \rho_i^{x_i}}{\delta_i} \prod_{j=1}^{y_i} q_i^2(j), \quad x_i \geq 0, y_i \geq 0, \forall i = 1, \dots, M,$$

and  $\prod_{j=1}^0 \equiv 1$ .

### 3 Simulations

In this section, we describe some numerical results for a supply system, depicted in Fig. 1: there are five nodes (working stations). Each node is interested by external flows of goods, while only nodes 1 and 2 are characterized by electrical signals sent by the CS. Following some fixed probabilities, unfinished goods can move from node  $i$  to node  $i + 1, i = 1, 2, 3, 4$ ; from node 5, parts either leave the system or come back to node 1. For electrical impulses, the dynamics is the same of the unfinished goods. Notice that such system describes the different construction steps of car engines in a real Italian company. For privacy reasons, names and/or details about the real functionalities of each node are avoided.

**Fig. 1** Topology of the supply system



Assume that:  $A^1_{0i} = 20 \forall i = 1, \dots, 5$ ,  $b^1_1 = 30$ ,  $b^1_2 = b^1_5 = 40$ ,  $b^1_3 = b^1_4 = 35$ , where all such quantities are seen as number of parts per minute;  $A^2_{01} = A^2_{02} = 5$ ,  $A^2_{03} = A^2_{04} = A^2_{05} = 0$ ;  $b^2_1 = 30$ ,  $b^2_2 = b^2_5 = 40$ ,  $b^2_3 = b^2_4 = 35$ ;  $b^2_1 = b^2_2 = b^2_3 = b^2_4 = 35$ ,  $b^2_5 = 30$ , where such last quantities are intended as number of control signals per minute. As for matrices  $\mathbf{G}^1$ ,  $\mathbf{H}^1$ ,  $\mathbf{G}^2$  and  $\mathbf{H}^2$ , they have zero elements with the following exceptions:  $\mathbf{G}^1(m, m + 1) = \mathbf{H}^1(m, m + 1) = \mathbf{G}^2(m, m + 1) = \mathbf{H}^2(m, m + 1) = 0.5 \forall m = 1, \dots, 4$ ;  $\mathbf{G}^1(5, 1) = \mathbf{H}^1(5, 1) = 0.2$ .

Table 1 reports some values of the stationary probabilities for node 2. The choice of considering such node is due to the fact that it is inside the network and interested by both external parts and control signals rates. If the number of control impulses grows,  $\pi_2(x_2, y_2)$  decreases. This occurs because controls in nodes determine variations of the ordinary parts dynamics, in terms either of movements to other nodes or possible destructions.

In order to analyze the behaviour of stationary probabilities versus the number of parts, we define the probability  $\tilde{\pi}_i(x_i) := \sum_{y_i=0}^{+\infty} \pi_i(x_i, y_i)$  that a node  $i$ ,  $i = 1, \dots, 5$ , has  $x_i$  goods. In Table 2, we have some values of  $\tilde{\pi}_1$  and  $\tilde{\pi}_2$ .

Notice that  $\tilde{\pi}_i(x_i)$  grows when the number of parts decreases and, moreover,  $\tilde{\pi}_2(x_2) > \tilde{\pi}_1(x_1)$ , indicating that node 2 tends to have more goods than node 1.

Further studies are done by computing the mean number of parts in the network, defined as:

$$N_p := \sum_{x_i=0}^{+\infty} x_i \left( \sum_{y_j=0}^{+\infty} y_j \pi_i(x_i, y_j) \right).$$

If we depict  $N$  versus  $A^1_{01}$  (Fig. 2, left) and versus  $A^2_{02}$  (Fig. 2, right), we get an idea of the ergodicity condition of the network process.

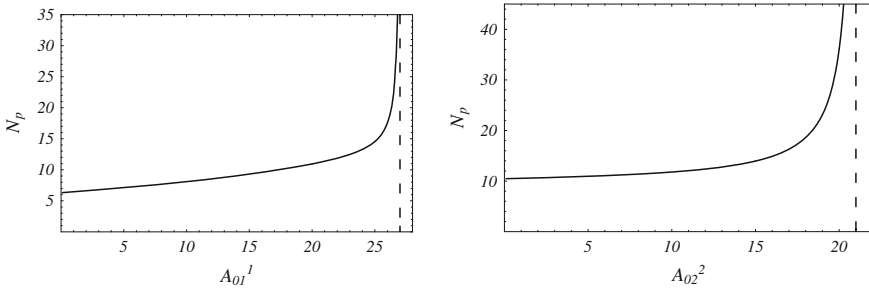
In particular, if the network is simulated with:

**Table 1** Values of  $\pi_2(x_2, y_2)$  for different values of  $x_2$  (columns) and  $y_2$  (rows)

$x_2 \setminus y_2$	1	2	3
1	0.0606621	0.0327992	0.01773410
2	0.0349247	0.0188833	0.01021000
3	0.0201070	0.0108716	0.00587813

**Table 2**  $\tilde{\pi}_i$  for node  $i$  (columns),  $i = 1, 2$ , assuming  $x_j$  unfinished goods (rows),  $j = 1, 2, 3$

$i \setminus x_j$	1	2	3
1	0.162585	0.129363	0.1029290
2	0.244266	0.140630	0.0809641



**Fig. 2**  $N_p$  versus  $A_{01}^1$  (left) and  $A_{02}^2$  (right)

- $A_{01}^1$  variable and other parameters equal to the ones used before, node 1 becomes instable when  $A_{01}^1 \simeq 27$ , leading to the instability of the whole network;
- $A_{02}^2$  variable and other parameters equal to the ones used before, the network process is not ergodic anymore if  $A_{02}^2 \geq 21$ .

Similar phenomena occur by analyzing the behaviour of  $N_p$  vs  $A_{02}^1$  and versus  $A_{01}^2$ .

## 4 Conclusions

In this paper, a queueing network, that models a real supply chain to assemble car engines, is described. For such a system, steady state probabilities have been computed in product form. Numerical tests have established that the control signals inside each node influence deeply the dynamics of the overall network.

Future work activities aim at applying the proposed model to scenarios, that involve other real supply systems in different contexts. In this direction, obvious modifications of the mathematical background foresee a fuzzy logic approach in order to obtain a robust optimization for the performances of the supply systems.

## References

1. Artalejo, J.R.: G-networks: a versatile approach for work removal in queueing networks. *Eur. J. Oper. Res.* **126**, 233–249 (2000)
2. Bocharov, P.P., D’Apice, C., Pechinkin, A.V., Salerno, S.: *Queueing Theory, Modern Probability and Statistics*. VSP, The Netherlands (2004)
3. Cutolo, A., Piccoli, B., Rarità, L.: An Upwind-Euler scheme for an ODE-PDE model of supply chains. *SIAM J. Comput.* **33**(4), 1669–1688 (2011)
4. De Falco, M., Gaeta, M., Loia, V., Rarità, L., Tomasiello, S.: Differential quadrature-based numerical solutions of a fluid dynamic model for supply chains. *Commun. Math. Sci.* **14**(5), 1467–1476 (2016)

5. Gelenbe, E.: Product form queueing networks with positive and negative customers. *J. Appl. Probab.* **28**, 656–663 (1991)
6. Gelenbe, E.: G-networks with triggered customer movement. *J. Appl. Prob.* **30**, 742–748 (1993)
7. Gelenbe, E., Pujolle, G.: *Introduction to Queueing Networks*, 2nd edn. Wiley, New York (1999)
8. Jackson, J.R.: Networks of waiting lines. *Oper. Res.* **5**, 518–521 (1957)
9. Pasquino, N., Rarità, L.: Automotive processes simulated by an ODE-PDE model. In: *Proceedings of EMSS 2012, 24th European Modeling and Simulation Symposium*, 19–21 Sept 2012, Vienna, Austria, pp. 352–361 (2012)
10. Tomasiello, S., Macías-Díaz, J.E.: Note on a picard-like method for caputo fuzzy fractional differential equations. *Appl. Math. Inform. Sci.* **11**(1), 281–287 (2017)
11. Yao, D.D., Buzacott, J.A.: The exponentialization approach to flexible manufacturing systems models with general processing times. *Eur. J. Oper. Res.* **24**, 410–416 (1986)

# Capital Asset Pricing Model—A Structured Robust Approach

Raquel J. Fonseca

**Abstract** We present an alternative robust formulation to the determination of the Capital Asset Pricing Model. We consider that both the asset and the market returns are uncertain and subject to some structured perturbations, and calculate a robust beta. The resulting structured robust model can be easily solved using semidefinite programming and is subsequently tested with data from the portuguese stock market. Numerical results have shown that the robust beta is, not only greater than the traditional least squares beta, but also that it tends to increase with a growing number of perturbation matrices, thus yielding a higher level of systematic risk.

**Keywords** Robust optimization • Capital asset pricing model • Linear regression

## 1 Introduction

Praised for its simplicity, but criticized by its lack of performance, the Capital Asset Pricing Model (CAPM) has been studied, tested and refuted by the academic community for many years. Yet it still remains as the most commonly used method to estimate the cost of capital, despite its difficulties, limitations and the poor correlation between the risk premiums, as Fernandez [5] so well described it.

First developed by Sharpe [9] and Lintner [8], the CAPM has been subject to numerous empirical tests with financial data from several different countries. Fama and French [4] provide an extensive review of the main CAPM theory and assumptions, early and recent empirical tests, limitations, as well as more suitable alternatives such as the Three-Factor-Model.

We propose the use of robust optimization techniques as developed by El Ghaoui and Lebret [3], where coefficient matrices in least-square problems are unknown but bounded. Their work is then extended by Hindi and Boyd [6]. Calafiore and Dabbene

---

R.J. Fonseca (✉)  
DEIO CMAFCIO Faculdade de Ciências, Universidade de Lisboa,  
1749-016 Lisboa, Portugal  
e-mail: rjfonseca@ciencias.ulisboa.pt

[1], and Calafiore et al. [2] have applied these novel methods to least square problems with stochastic uncertainty and to worst case residual minimization, and Vanli et al. [11] on regret minimization.

The main contributions of this study may be summarized as follows:

- We apply a robust optimization technique to a financial pricing model, by developing a structured model as in [3]. Although robust optimization techniques have been exhaustively applied to portfolio optimization problems, no research has been conducted, to our knowledge, on the determination of a worst case beta in accordance with the CAPM;
- Based on financial historical data from the portuguese stock market, we estimate the historical betas corresponding to the robust structured case, and compare them to the regular, non robust case;
- We estimate the Security Market Line (SML) given the least squares beta and the robust beta, concluding on the better results of the latter in terms of data fitting.

The next section briefly introduces the CAPM and its main assumptions, while in Sect. 3 we present the robust counterpart of the model, as well as the uncertainty set for the coefficient matrices. Section 4 describes some of the numerical results obtained with real data from the portuguese stock market. We conclude in Sect. 5.

## 2 The Capital Asset Pricing Model

The CAPM as proposed by Sharpe [9] and Lintner [8] describes a linear relationship between the asset risk premium and the market risk premium:

$$E(r) - r_f = \beta[E(r_M) - r_f], \tag{1}$$

where  $E(r)$  and  $E(r_M)$  stand for the expected asset and market returns, respectively, and  $r_f$  is the risk free rate. The goal is to find the parameter  $\beta$  representing the systematic risk of the asset, that is, risk that can not be diversified. Estimation of  $\beta$  usually entails minimizing the sum of squared errors,  $\sum_{i=1}^n |e_i|^2$ , of:

$$(r - r_f) = \alpha + \beta(r_M - r_f) + e_i, \tag{2}$$

over a set of  $n$  historical observations. Throughout the manuscript, we reverse to the most commonly used matrix notation, with:

$$\min_{\alpha, \beta} \sum_{i=1}^n |e_i|^2 = |[\alpha + \beta(r_M - r_f)] - (r - r_f)|^2 \tag{3}$$

$$= ||Ax - b||^2, \tag{4}$$

and solution given by:  $x = (A^T A)^{-1} A^T b$ . In the next section, we try to improve on these results by applying worst case scenario techniques.

### 3 The Robust Counterpart

Let us now assume that both matrices  $A$  and  $b$  are subject to perturbations, and that we do not know their exact value, or are unable to estimate them properly. From CAPM main assumptions, only the asset and the market returns are subject to perturbations. The risk free rate, by its implicit nature, is not subject to any risk and its value is known exactly. While finding an appropriate proxy for the risk free asset is one of the limitations of the CAPM, the main problem, however, derives from the relationship between the asset and the market returns.

We consider a perturbation  $\delta$  and the following matrices:  $A_0, \dots, A_L \in \mathbb{R}^{n \times m}$ , and  $b_0, \dots, b_L \in \mathbb{R}^n$ , such that the uncertainty set  $\Xi$  may be written as:

$$\Xi = \left\{ [\delta \ \Delta A \ \Delta b] : \Delta A = \sum_{i=1}^L \delta_i A_i, \ \Delta b = \sum_{i=1}^L \delta_i b_i, \ \|\delta\|_p \leq \rho \right\} \tag{5}$$

Matrices  $A_0$  and  $b_0$  correspond to our original matrices, which include the measured or estimated data. Parameter  $p$  states the norm over which the uncertainty set is built. Unless otherwise stated, this will be the Euclidean norm, therefore  $p = 2$ .

The worst case residual may be determined by the following formulation:

$$\min_x \max_{[\Delta A \ \Delta b] \in \Xi} \|(A_0 + \Delta A)x - (b_0 + \Delta b)\|_2 \tag{6}$$

Using semidefinite programming, we may solve problem (6) to optimality. We start by defining a new matrix  $M$ :

$$M(x) = [A_1 x - b_1, \dots, A_L x - b_L]$$

Additionally:

$$F = M(x)^T M(x), \ g = M(x)^T (A_0 x - b_0), \ \text{and} \ h = \|A_0 x - b_0\|_2$$

Applying this new notation, the structured problem (6) may be written as:

$$\min_x \max_{\|\delta\|_2 \leq \rho} \begin{bmatrix} 1 \\ \delta \end{bmatrix}^T \begin{bmatrix} h & g^T \\ g & F \end{bmatrix} \begin{bmatrix} 1 \\ \delta \end{bmatrix}$$

*Proof* Let the inner maximization problem be written as:

$$\begin{aligned} \max_{\|\delta\|_2 \leq \rho} & \begin{bmatrix} h + \delta g & g^T + \delta F \end{bmatrix} \begin{bmatrix} 1 \\ \delta \end{bmatrix} \\ & = h + \delta^T g + (g^T + \delta^T F)\delta \\ & = h + \delta^T g + g^T \delta + \delta^T F \delta. \end{aligned}$$

After replacing the above expression with the newly defined notation:

$$\begin{aligned}
 & \max_{\|\delta\|_2 \leq \rho} \|A_0x - b_0\|_2 + \delta^T M(x)^T (A_0x - b_0) + [M(x)^T (A_0x - b_0)]^T \delta + \delta^T M(x)^T M(x) \delta \\
 & = \|A_0x - b_0 + \delta^T M(x)\|_2 \\
 & = \|A_0x - b_0 + \delta^T [A_1x - b_1, \dots, A_px - b_L]\|_2 \\
 & = \|A_0x - b_0 + \sum_{i=1}^L \delta_i (A_i x - b_i)\|_2,
 \end{aligned}$$

we arrive at our original robust structured problem (6).

We continue to simplify our model by stating that:

$$\min_{x, \lambda} \lambda, \quad \text{s.t.} \quad \begin{bmatrix} 1 \\ \delta \end{bmatrix}^T \begin{bmatrix} \lambda - h & -g^T \\ -g & -F \end{bmatrix} \begin{bmatrix} 1 \\ \delta \end{bmatrix} \geq 0, \quad \forall \delta \in \Xi \tag{7}$$

**Theorem 1** (*S*-lemma) *Given two symmetric matrices  $\mathcal{W}$  and  $\mathcal{S}$  of the same size and assuming the inequality  $\xi^T \mathcal{W} \xi \geq 0$  is strictly feasible, that is,  $\bar{\xi}^T \mathcal{W} \bar{\xi} > 0$  for some  $\bar{\xi} \in \mathbb{R}^k$ , then the following equivalence holds:*

$$[\xi^T \mathcal{W} \xi \geq 0 \Rightarrow \xi^T \mathcal{S} \xi \geq 0] \Leftrightarrow \exists \tau \geq 0 : \mathcal{S} \geq \tau \mathcal{W}. \tag{8}$$

Relationship  $\mathcal{S} \geq \tau \mathcal{W}$  indicates the matrix  $(\mathcal{S} - \tau \mathcal{W})$  is positive semidefinite.

Given that:

$$\mathcal{W} = \begin{bmatrix} \rho & 0 \\ 0 & -1 \end{bmatrix}, \quad \mathcal{S} = \begin{bmatrix} \lambda - h & -g^T \\ -g & -F \end{bmatrix}, \quad \text{and} \quad \xi = \delta = \begin{bmatrix} 1 \\ \delta \end{bmatrix},$$

we conclude, by applying the *S*-lemma, that there exists some  $\tau \geq 0$ , such that:

$$\begin{aligned}
 & \mathcal{S} \geq \tau \mathcal{W} \\
 & \begin{bmatrix} \lambda - h & -g^T \\ -g & -F \end{bmatrix} \geq \tau \begin{bmatrix} \rho & 0 \\ 0 & -1 \end{bmatrix} \\
 & \begin{bmatrix} \lambda - h - \rho\tau & -g^T \\ -g & \tau \mathcal{S} - F \end{bmatrix} \geq 0
 \end{aligned}$$

We re-write problem (7) as:

$$\min_{x, \lambda, \tau} \lambda, \quad \text{s.t.} \quad \begin{bmatrix} \lambda - h - \rho\tau & -g^T \\ -g & \tau \mathcal{S} - F \end{bmatrix} \geq 0, \tag{9}$$



whose solution may be found by solving the following semidefinite problem (SDP) using interior-point methods:

$$\min_{x, \lambda, \tau} \lambda, \quad \text{s.t.} \quad \begin{bmatrix} \lambda - \rho\tau & 0 & (A_0x - b_0)^T \\ 0 & \tau \mathcal{J} & M^T \\ (A_0x - b_0) & M & \mathcal{J} \end{bmatrix} \geq 0 \tag{10}$$

*Proof* By using Schur complements in constraint (10) above, we show that:

$$\begin{aligned} & \begin{bmatrix} \lambda - \rho\tau & 0 \\ 0 & \tau \mathcal{J} \end{bmatrix} - \begin{bmatrix} (A_0x - b_0)^T \\ M^T \end{bmatrix} [\mathcal{J}^{-1}] \begin{bmatrix} (A_0x - b_0) & M \end{bmatrix} \geq 0 \\ & \begin{bmatrix} \lambda - \rho\tau & 0 \\ 0 & \tau \mathcal{J} \end{bmatrix} - \begin{bmatrix} (A_0x - b_0)^T (A_0x - b_0) & (A_0x - b_0)^T M \\ M^T (A_0x - b_0) & M^T M \end{bmatrix} \geq 0 \\ & \begin{bmatrix} \lambda - \rho\tau - (A_0x - b_0)^T (A_0x - b_0) & -(A_0x - b_0)^T M \\ -M^T (A_0x - b_0) & \tau \mathcal{J} - M^T M \end{bmatrix} \geq 0 \end{aligned}$$

Note how, by replacing with the notation previously defined, this formulation is equivalent to our original problem (9).

### 4 Numerical Results

In simulating the perturbation matrices, we follow a similar approach to Kuhn et al. [7], in which portfolio wealth is maximized for the worst case returns. The nominal matrices  $A_0$  and  $b_0$  are subject to some perturbations, determined at the same time by the standard deviation of the returns and by a second parameter  $D$ , that measures the distance from the nominal value in terms of the standard deviation. In our computational experiments, parameter  $D$  was set to any value between  $-2$  and  $2$ . This ensures that not all of the returns are wrongly estimated at the same time, since in some cases  $D$  will take the value  $0$ .

From the uncertainty set  $\Xi$  defined in (5), we have:

$$\Delta A = \sigma_A \sum_{i=1}^L \delta_i D_i, \quad \text{and} \quad \Delta b = \sigma_b \sum_{i=1}^L \delta_i D_i.$$

For simplicity, we assume matrices  $D_i$  are the same for both uncertain matrices  $A_i$  and  $b_i$ , but this does not need to be the case.

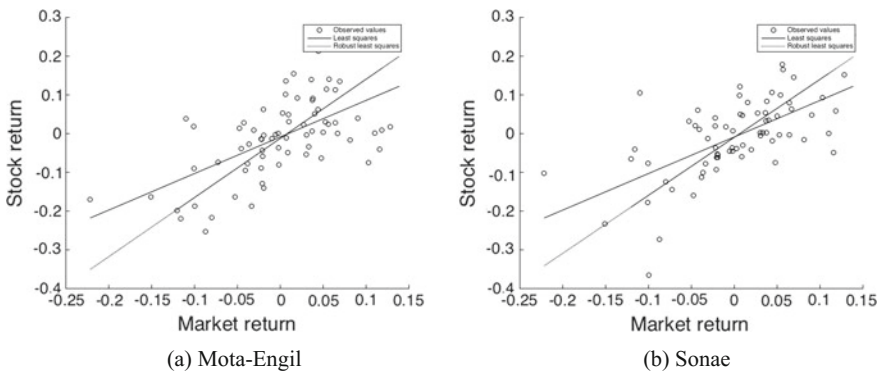
In order to run the structured robust model, monthly historical returns from January 2006 to December 2011 were collected, with a total of 72 observations. As asset returns, we used the returns from 12 of the companies included in the portuguese stock market index, PSI-20 (not all companies were included as some did not have available data as early as 2006). The European market risk premium was taken from

the data library of Fama and French, defined as “the return on a region’s value-weight market portfolio minus the U.S. 1 month T-bill rate.” This index includes equity from the following european countries: Austria, Belgium, Switzerland, Germany, Denmark, Spain, Finland, France, Great Britain, Greece, Ireland, Italy, Netherlands, Norway, Portugal and Sweden.

As a starting point, the robust model was ran with parameter  $\rho$  equal to one and ten perturbation matrices  $D$ , using the solver SDPT3 [10]. Figure 1 shows preliminary results from this model and compares the least squares beta and the robust structured beta for two of the studied companies, Mota-Engil and Sonae, chosen by the highest coefficient of determination.

For comparison purposes, Table 1 shows the least squares beta and the structured robust beta for the same two companies considered. The robust beta was determined for a growing number of perturbation matrices in order to conclude about this relationship. The  $R^2$  for the least squares model is also shown as a measure of the adequacy of the model. As expected, given other empirical tests of the CAPM, its value tends to be quite low, below 40%.

From Table 1, we are drawn to conclude that the robust beta tends to be higher than the least squares beta, meaning that the level of systematic or non-diversifiable risk has increased. This is consistent with the premisses of robust optimization: since



**Fig. 1** Least squares and robust structured betas

**Table 1** Least squares and robust betas with growing perturbations

Companies	Mota-Engil	Sonae
$R^2$	0.3733	0.3975
Least squares	0.9331	0.9439
D = 3	1.4185	1.4632
D = 5	1.4004	1.4942
D = 10	1.5054	1.3960
D = 15	1.5273	1.4795

the investor is accounting for the worst case scenario, she will tend to be more conservative and estimate a higher risk level.

Secondly, as the number of perturbation matrices increases, so does the robust beta. Again, an increase in the number of perturbations leads to an enlargement of the uncertainty set, and the consideration of a greater number of asset and market returns in the optimization problem. Optimizing for the worst case scenario will therefore result in an increased beta. Additionally, the robust beta has remained positive in all of the model runs, not questioning the fact, therefore, that these are risky assets.

To assess the predictive power of the robust beta, we conducted a preliminary ex-post analysis. Figure 2 plots the annualized average asset risk premium from 2012 to 2015 for the twelve companies included in the study, against the least squares and the robust beta. The later points represent a shift towards the right hand side of the graph as the robust beta is always greater than the least squares beta. The Security Market Line (SML) is obtained for both cases by linear regression techniques.

We can directly observe that the robust SML (based on the robust betas) is flatter than the least squares SML, confirming some of the previous findings in the area regarding the long run SML [4]. In terms of goodness of fit measured by the coefficient of determination, the regression based on the robust betas has a greater  $R^2$ , 22.4% compared to 20.9% of the least squares beta. Results are affected by the presence of an outlier: company Pharol, the only one with a negative annualized average return of 38.9% and a least squares and a robust beta of 0.29 and 1.04, respectively. Without this observation, while the robust  $R^2$  decreases to 20%, the least squares  $R^2$  drops to zero, accentuating the difference between the two betas and the better results of the robust beta.

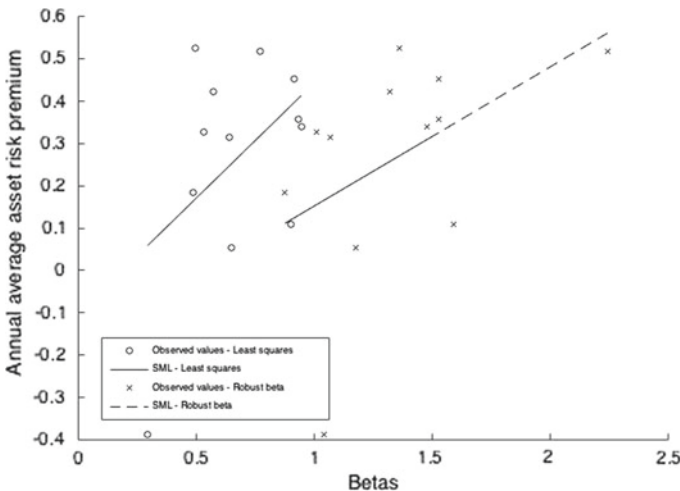


Fig. 2 Ex-post analysis: security market line

## 5 Conclusion

From the preliminary tests described in the previous section, we tend to conclude that not only the robust beta is greater than the least squares beta, but also that it yields better results in terms of predictive power. The SML estimated through linear regression provides a better fit to the observed values when the robust betas are used, and it appears to be flatter, which was one of the limitations attributed to ordinary least squares estimation of betas from previous studies. These results are therefore motivating in order to extend this approach to the estimation of portfolio betas. The use of robust optimization techniques in portfolio optimization problems stems from the minimum return guarantees provided. Further research on whether similar guarantees may be given in these type of problems still needs to be accomplished.

**Acknowledgements** This work is supported by National Funding from FCT—Fundação para a Ciência e a Tecnologia, under the project: UID/MAT/04561/2013.

## References

1. Calafiore, G., Dabbene, F.: Near optimal solutions to least-squares problems with stochastic uncertainty. *Syst. Control Lett.* **54**, 1219–1232 (2005)
2. Calafiore, G.C., Topcu, U., El-Ghaoui, L.: Parameter estimation with expected and residual-at-risk criteria. *Syst. Control Lett.* **58**(1), 39–46 (2009)
3. El Ghaoui, L., Lebret, H.: Robust solutions to least-squares problems with uncertain data. *SIAM J. Matrix Anal. Appl.* **18**(4), 1035–1064 (1997)
4. Fama, E.F., French, K.R.: The capital asset pricing model: theory and evidence. *J. Econ. Perspect.* **18**(3), 25–46 (2004)
5. Fernandez, P.: CAPM: an absurd model. *Bus. Valuati. Rev.* **34**(1), 4–23 (2015)
6. Hindi, H., Boyd, S.: Robust solutions to  $l_1$ ,  $l_2$ , and  $l$  infinity uncertain linear approximation problems using convex optimization. *Am. Control Conf.* **6**, 3487–3491 (1998)
7. Kuhn, D., Parpas, P., Rustem, B., Fonseca, R.: Dynamic mean-variance portfolio analysis under model risk. *J. Comput. Financ.* **12**(4), 91–115 (2009)
8. Lintner, J.: The valuation of risk assets and the selection of risky investments in stock portfolios and capital budgets. *Rev. Econ. Stat.* **47**(1), 13–37 (1965)
9. Sharpe, W.F.: Capital asset prices: a theory of market equilibrium under conditions of risk. *J. Financ.* **19**(3), 425–442 (1964)
10. Tutuncu, R.H., Toh, K.C., Todd, M.J.: Solving semidefinite-quadratic-linear programs using SDPT3. *Math. Program.* **95**(2), 189–217 (2003)
11. Vanli, N.D., Donmez, M.A., Kozat, S.S.: Robust least squares methods under bounded data uncertainties. *Digit. Signal Process.* **36**, 82–92 (2015)

# On the Properties of Interval Linear Programs with a Fixed Coefficient Matrix

Elif Garajová, Milan Hladík and Miroslav Rada

**Abstract** Interval programming is a modern tool for dealing with uncertainty in practical optimization problems. In this paper, we consider a special class of interval linear programs with interval coefficients occurring only in the objective function and the right-hand-side vector, i.e. programs with a fixed (real) coefficient matrix. The main focus of the paper is on the complexity-theoretic properties of interval linear programs. We study the problems of testing weak and strong feasibility, unboundedness and optimality of an interval linear program with a fixed coefficient matrix. While some of these hard decision problems become solvable in polynomial time, many remain (co-)NP-hard even in this special case. Namely, we prove that testing strong feasibility, unboundedness and optimality remains co-NP-hard for programs described by equations with non-negative variables, while all of the weak properties are easy to decide. For inequality-constrained programs, the (co-)NP-hardness results hold for the problems of testing weak unboundedness and strong optimality. However, if we also require all variables of the inequality-constrained program to be non-negative, all of the discussed problems are easy to decide.

**Keywords** Interval linear programming · Computational complexity

---

E. Garajová (✉) · M. Hladík  
Faculty of Mathematics and Physics, Department of Applied Mathematics,  
Charles University, Malostranské nám. 25, 11800 Prague, Czech Republic  
e-mail: elif@kam.mff.cuni.cz

M. Hladík  
e-mail: hladik@kam.mff.cuni.cz

M. Rada  
Department of Financial Accounting and Auditing, University of Economics,  
Prague, nám. W. Churchilla 4, 13067 Prague, Czech Republic  
e-mail: miroslav.rada@vse.cz

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_40

## 1 Introduction

Practical optimization problems are often flawed by uncertainty or inexactness due to errors in measurements, approximations and estimations. To create an accurate mathematical representation of the real world, these inaccuracies need to be considered and included in the model. Many different approaches to modeling uncertainty in optimization have emerged throughout the last years, such as robust optimization, stochastic programming or parametric programming.

In this paper, we assume the form of interval-valued coefficients enclosing the exact data. We adopt the approach of so-called interval linear programming, which can be seen as a special case of multiparametric programming. Unlike stochastic programming, we have no further assumptions on the probability distribution of the given data. While some of the concepts used in interval programming are similar to those of robust optimization [1], we do not necessarily assume the worst-case realization of the data—we are interested in the properties of all possible realizations. Also, we focus mainly on the properties of the uncertain program as a whole, rather than studying the properties of some stable feasible or optimal solutions.

The paper discusses the complexity properties of a special class of interval linear programs, in which only the objective function and the right-hand side are affected by uncertainty. We build on a survey by Hladík [5] providing a summary of the complexity results for the general case. Obviously, all problems solvable in polynomial time remain polynomial for the special case. For the hard problems, the situation is different. Since most of the referenced NP-hardness proofs are based on programs with an interval coefficient matrix, we cannot directly infer any results about the complexity of the harder problems in our special case.

## 2 Interval Linear Programming

Let us introduce some notation and conventions: For a vector  $v \in \mathbb{R}^n$  we denote by  $\text{diag}(v)$  the diagonal matrix with entries  $\text{diag}(v)_{ii} = v_i$  for  $i \in \{1, \dots, n\}$ . The symbol  $e$  denotes the all-ones vector  $(1, \dots, 1)^T$ . The inequality relations on the set of matrices, as well as the absolute value operator  $|\cdot|$ , are understood entry-wise.

Given two matrices  $\underline{A}, \bar{A} \in \mathbb{R}^{m \times n}$ , we define the *interval matrix*  $\mathbf{A} = [\underline{A}, \bar{A}]$  as the set of matrices  $\{A \in \mathbb{R}^{m \times n} : \underline{A} \leq A \leq \bar{A}\}$ , where the matrices  $\underline{A}, \bar{A}$  are the *lower* and *upper bound* of  $\mathbf{A}$ , respectively. An interval matrix can be also represented by the *center*  $A_c = 1/2(\bar{A} + \underline{A})$  and *radius*  $A_\Delta = 1/2(\bar{A} - \underline{A})$ . Interval vectors and matrices are distinguished from reals by bold letters. The set of all  $m \times n$  interval matrices will be denoted by  $\mathbb{IR}^{m \times n}$ , the symbol  $\mathbb{IR}^n$  denotes the set of  $n$ -dimensional interval vectors.

Let  $\mathbf{A} \in \mathbb{IR}^{m \times n}$ ,  $\mathbf{b} \in \mathbb{IR}^m$  and  $\mathbf{c} \in \mathbb{IR}^n$  be given. An *interval linear program* (ILP) is then defined as the family of linear programs

$$\text{minimize } \mathbf{c}^T x \text{ subject to } x \in \mathcal{M}(A, b), \tag{1}$$

where  $A \in \mathbf{A}, b \in \mathbf{b}, c \in \mathbf{c}$  and  $\mathcal{M}(A, b)$  denotes the feasible set. For the sake of notational simplicity, we will write problem (1) shortly as

$$\text{minimize } \mathbf{c}^T x \text{ subject to } x \in \mathcal{M}(\mathbf{A}, \mathbf{b}). \tag{2}$$

A particular linear program in an ILP is called a *scenario*. If some of the coefficients are fixed, we also use the term “scenario” to refer to a choice of the interval coefficients, which uniquely determine the program. A vector  $x \in \mathbb{R}^n$  is a *weakly feasible (optimal)* solution of (2), if it is a feasible (optimal) solution of some scenario.

In this paper, we consider the special case of programs with a fixed coefficient matrix  $A \in \mathbb{R}^{m \times n}$ , interval right-hand side  $\mathbf{b} \in \mathbb{IR}^m$  and interval objective  $\mathbf{c} \in \mathbb{IR}^n$ . We assume that the program is given in one of the following commonly used forms:

$$\text{minimize } \mathbf{c}^T x \text{ subject to } Ax = \mathbf{b}, x \geq 0, \tag{I}$$

$$\text{minimize } \mathbf{c}^T x \text{ subject to } Ax \leq \mathbf{b}, \tag{II}$$

$$\text{minimize } \mathbf{c}^T x \text{ subject to } Ax \leq \mathbf{b}, x \geq 0. \tag{III}$$

In classical linear programming, programs can be equivalently restated in any of the forms using the standard transformations. However, this is not always possible for ILPs, therefore, we need to study programs in various forms separately.

### 3 Properties of Interval Linear Programs

We will now explore the complexity of some decision problems related to the properties of interval linear programs with a fixed coefficient matrix. The three main properties that will be considered are:

- **Feasibility:** Are all/some scenarios feasible?
- **Unboundedness:** Do all/some scenarios have an unbounded objective function?
- **Optimality:** Do all/some scenarios possess an optimal solution?

An ILP is called *weakly feasible (optimal)*, if there exists a scenario having a feasible (optimal) solution. An ILP is *weakly unbounded*, if there exists a feasible scenario with an unbounded objective function. Similarly, it is *strongly feasible/optimal/unbounded*, if the corresponding property holds for each scenario.

These properties have been studied mostly for general intervals programs, see Table 1 for an overview of the results based on survey [5]. In the rest of the paper, we will show that for programs with a fixed matrix, the complexity properties summarized in Table 2 hold (the changes compared to Table 1 are emphasized).

Another interesting problem in interval optimization is the computation of lower and upper bounds on the optimal value of the objective function, or the *optimal*

**Table 1** Complexity of testing properties of general ILPs

	$\min \mathbf{c}^T x$ $Ax = \mathbf{b}, x \geq 0$	$\min \mathbf{c}^T x$ $Ax \leq \mathbf{b}$	$\min \mathbf{c}^T x$ $Ax \leq \mathbf{b}, x \geq 0$
Strong feasibility	co-NP-hard [10]	Polynomial [10]	Polynomial [10]
Weak feasibility	Polynomial [10]	NP-hard [10]	Polynomial [10]
Strong unboundedness	co-NP-hard [7]	Polynomial [5]	Polynomial [5]
Weak unboundedness	Open	NP-hard <sup>1</sup>	Polynomial [5]
Strong optimality	co-NP-hard [9]	co-NP-hard <sup>2</sup>	Polynomial [5]
Weak optimality	NP-hard <sup>3</sup>	NP-hard <sup>3</sup>	Open

<sup>1</sup> Follows from NP-hardness of testing weak feasibility using the construction in Theorem 3

<sup>2</sup> Follows from the result for type (I) by duality

<sup>3</sup> Follows from NP-hardness of testing weak feasibility for type (II), then by duality for type (I)

**Table 2** Complexity of testing properties of ILPs with a fixed coefficient matrix

	$\min \mathbf{c}^T x$ $Ax = \mathbf{b}, x \geq 0$	$\min \mathbf{c}^T x$ $Ax \leq \mathbf{b}$	$\min \mathbf{c}^T x$ $Ax \leq \mathbf{b}, x \geq 0$
Strong feasibility	co-NP-hard	Polynomial	Polynomial
Weak feasibility	Polynomial	<i>Polynomial</i>	Polynomial
Strong unboundedness	co-NP-hard	Polynomial	Polynomial
Weak unboundedness	<i>Polynomial</i>	NP-hard	Polynomial
Strong optimality	co-NP-hard	co-NP-hard	Polynomial
Weak optimality	<i>Polynomial</i>	<i>Polynomial</i>	<i>Polynomial</i>

*value range*. For the main results on the optimal value range, we refer the reader to an overview by Rohn [9]. Since the corresponding NP-hardness proof is already based on a problem with a fixed matrix, the results also hold for the special case considered here.

Regarding the set of optimal solutions, there are two interesting questions often discussed in interval linear programming: describing the set of all optimal solutions and determining, whether a given solution is (weakly) optimal. The former problem was studied in the thesis of the first author [2], where an exponential description of the optimal set is given for programs with a fixed matrix. For the latter problem, a polynomial-time algorithm was proposed by Li et al. [8] for linear programs with an interval right-hand side.

### 3.1 Feasibility

Feasibility of interval linear systems was studied by Rohn [10] and the results were generalized by Hladík [6] to mixed systems. Rohn proved that testing strong feasibility of systems of equations with non-negative variables is co-NP-hard and testing



weak feasibility of systems of inequalities is NP-hard. Theorem 2 shows that the former result remains true even in our special case.

Let us now prove an auxiliary result, which will be used for reductions in the upcoming NP-hardness proofs:

**Theorem 1** *Testing weak feasibility is NP-hard for interval systems in the form*

$$Ax \leq 0, \mathbf{b}^T x < 0. \tag{3}$$

*Proof* By the results of Hladík [4], we have that checking feasibility of the system

$$|Ax| \leq e, e^T|x| > 1 \tag{4}$$

is an NP-hard problem. Let us now show that checking feasibility of system (4) is equivalent to checking feasibility of the system

$$|Ax| \leq ey, y \geq 0, e^T|x| > y. \tag{5}$$

Obviously, if  $x$  is a feasible solution of (4), then the pair  $(x, 1)$  solves system (5). Conversely, let  $(x, y)$  be a solution of (5). If  $y > 0$ , then the vector  $x/y$  is a solution of system (4). For  $y = 0$  we have a solution  $x$  of system (5) satisfying  $Ax = 0, e^T|x| > 0$ . Consider the vector  $x' = \frac{x}{e^T|x| - \epsilon}$  for some  $\epsilon$  with  $0 < \epsilon < e^T|x|$ . Then  $x'$  solves system (4), since  $|Ax'| = 0 \leq e$  and

$$e^T|x'| = e^T \left| \frac{x}{e^T|x| - \epsilon} \right| = \frac{e^T|x|}{e^T|x| - \epsilon} > 1.$$

By a variant of the Gerlach theorem [3], the inequality  $e^T|x| - y > 0$  is feasible if and only if the interval inequality  $[-e, e]^T x + y < 0$  is weakly feasible. This finishes the proof, since we have shown that it is NP-hard to test feasibility of systems in the form  $Ax - ey \leq 0, -Ax - ey \leq 0, -y \leq 0, [-e, e]^T x + y < 0$ .  $\square$

In the considered special case, testing strong feasibility of ILPs of the corresponding types can be performed by testing feasibility of the systems (see [10]):

- (I)  $Ax = b_c + \text{diag}(p)b_d, x \geq 0$  for each  $p \in \{\pm 1\}^m$ ,
- (II)  $Ax \leq \underline{b}$ ,
- (III)  $Ax \leq \underline{b}, x \geq 0$ .

The following theorem justifies the exponential condition for programs of type (I).

**Theorem 2** *Testing strong feasibility is co-NP-hard for interval linear systems of type  $Ax = b, x \geq 0$ .*

*Proof* By the Farkas lemma, the system  $Ax = b, x \geq 0$  is infeasible if and only if the system  $A^T y \geq 0, b^T y < 0$  is feasible. Therefore, an ILP of type (I) is weakly infeasible if and only if the interval system

$$A^T y \geq 0, \mathbf{b}^T y < 0 \tag{6}$$

is weakly feasible. However, by Theorem 1, this is an NP-hard decision problem.  $\square$

On the other hand, weak feasibility can be tested in polynomial time for all types of problems with a fixed matrix by testing feasibility of the following linear systems:

- (I)  $\underline{b} \leq Ax \leq \bar{b}, x \geq 0,$
- (II)  $Ax \leq \bar{b},$
- (III)  $Ax \leq \bar{b}, x \geq 0.$

### 3.2 Unboundedness

Strong unboundedness of interval linear programs was studied by Koníčková [7]. She proved that testing the property is co-NP-hard for general problems of type (I). By Theorem 3, this also holds for problems with a fixed coefficient matrix.

**Theorem 3** *Testing strong unboundedness is co-NP-hard for ILPs of type (I).*

*Proof* We construct a reduction from the strong feasibility problem, which is known to be co-NP-hard by Theorem 2. Given an interval linear system in the form  $Ax = \mathbf{b}, x \geq 0,$  consider the ILP

$$\text{maximize } z \text{ subject to } Ax = \mathbf{b}, x \geq 0, z \geq 0. \tag{7}$$

If the system  $Ax = b, x \geq 0$  is feasible for some  $b \in \mathbf{b},$  then the corresponding linear program in (7) is unbounded, since it is feasible and the value of  $z$  increases above all bounds. Therefore, if the interval system  $Ax = \mathbf{b}, x \geq 0$  is strongly feasible, then each scenario of ILP (7) is unbounded and the program is strongly unbounded. Conversely, if (7) is strongly unbounded, then it is by definition also strongly feasible. This shows that problem (7), which is an ILP of type (I), is strongly unbounded if and only if the corresponding interval system  $Ax = \mathbf{b}, x \geq 0$  is strongly feasible.  $\square$

To characterize strong unboundedness of an ILP, we can use unboundedness of the following linear programs (see [7]):

- (I) minimize  $\bar{c}^T x$  subject to  $Ax = b_c + \text{diag}(p)b_\Delta, x \geq 0$  for each  $p \in \{\pm 1\}^m,$
- (II) minimize  $\bar{c}^T x^1 - \underline{c}^T x^2$  subject to  $A(x^1 - x^2) \leq \underline{b}, x^1 \geq 0, x^2 \geq 0,$
- (III) minimize  $\bar{c}^T x$  subject to  $Ax \leq \underline{b}, x \geq 0.$

Unfortunately, little is known about weak unboundedness in the case of general interval linear programs. It is easy to see that in the special case with a fixed matrix, testing weak unboundedness can be done in polynomial time for problems of type (I) and (III). Non-negativity of the variables is crucial in this case, as illustrated by Theorem 4. To test weak unboundedness of an ILP with a fixed matrix, we can check weak feasibility of the following systems:

- (I)  $\underline{b} \leq Ax \leq \bar{b}, x \geq 0, Ad = 0, d \geq 0, \underline{c}^T d \leq -1,$
- (II)  $Ax \leq \bar{b}, Ad \leq 0, (c_c^T - c_d^T \text{diag}(p))x \leq -1$  for some  $p \in \{\pm 1\}^n,$
- (III)  $Ax \leq \bar{b}, x \geq 0, Ad \leq 0, d \geq 0, \underline{c}^T d \leq -1.$

**Theorem 4** *Testing weak unboundedness is NP-hard for ILPs of type (II).*

*Proof* An ILP of type (II) is weakly unbounded if and only if the system

$$Ax \leq \bar{b}, Ad \leq 0, \mathbf{c}^T d < 0 \tag{8}$$

is weakly feasible, i.e. there exists a feasible scenario with an unbounded direction, along which the objective function decreases. Since we know that testing weak feasibility of the subsystem  $Ad \leq 0, \mathbf{c}^T d < 0$  is NP-hard, in general (see Theorem 1), this also holds for the problem of testing weak unboundedness of (II).  $\square$

### 3.3 Optimality

The problem of testing strong optimality of an interval linear program was studied as the finite range problem (deciding finiteness of the lower and upper bound on the optimal objective value) by Rohn in [9]. The original theorem gives a co-NP-hardness result for equation-constrained programs with non-negative variables and an interval coefficient matrix. Theorem 5 provides a strengthening of the original results.

**Theorem 5** *Testing strong optimality is co-NP-hard for ILPs of types (I) and (II).*

*Proof* First, note that an ILP is strongly optimal if and only if the ILP formed by the dual linear programs is strongly optimal. Since the dual of an interval program of type (II) can be directly transformed into form (I), it is sufficient to prove the result for programs of type (I).

Again, we show a simple reduction from the strong feasibility problem. For an interval system  $Ax = \mathbf{b}, x \geq 0$  consider the program

$$\text{minimize } 0^T x \text{ subject to } Ax = \mathbf{b}, x \geq 0. \tag{9}$$

It is easy to see that program (9) is strongly optimal if and only if the corresponding system of type (I) is strongly feasible.  $\square$

Since strong optimality requires that both the primal and the dual ILP are strongly feasible, testing strong optimality can be performed using the conditions stated in Sect. 3.1 by testing feasibility of the systems:

- (I)  $Ax = b_c + \text{diag}(p)b_\Delta$ ,  $x \geq 0$ ,  $A^T y \leq \underline{c}$  for each  $p \in \{\pm 1\}^m$ ,  
 (II)  $Ax \leq \underline{b}$ ,  $A^T y = c_c + \text{diag}(p)c_\Delta$ ,  $y \leq 0$  for each  $p \in \{\pm 1\}^n$ ,  
 (III)  $Ax \leq \underline{b}$ ,  $x \geq 0$ ,  $A^T y \leq \underline{c}$ ,  $y \leq 0$ .

In general, it is hard to determine whether a given ILP is weakly optimal. However, contrary to the general case, we can test weak optimality of a program with a fixed matrix in polynomial time. By duality, this only requires to find a scenario, for which both the primal and the dual program are feasible.

Since we have a fixed coefficient matrix, the scenarios of the interval systems are only determined by the independent choice of  $b \in \mathbf{b}$  for the primal and  $c \in \mathbf{c}$  for the dual problem. This reduces the problem to testing weak feasibility of the primal and dual ILP, which is easy to test for all types of program. Using this approach, we need to test feasibility of the following systems:

- (I)  $\underline{b} \leq Ax \leq \bar{b}$ ,  $x \geq 0$ ,  $A^T y \leq \bar{c}$ ,  
 (II)  $Ax \leq \bar{b}$ ,  $\underline{c} \leq A^T y \leq \bar{c}$ ,  $y \leq 0$ ,  
 (III)  $Ax \leq \bar{b}$ ,  $x \geq 0$ ,  $A^T y \leq \bar{c}$ ,  $y \leq 0$ .

## 4 Conclusion

We have explored the decision problems related to the properties of interval linear programs on the special class of programs with a fixed coefficient matrix. Namely, we have addressed the problems of testing (weak and strong) feasibility, optimality and unboundedness of the program. While the restriction of the coefficients to real matrices may seem like a strong assumption, we have shown that many of the decision problems remain hard to decide even in this special case.

The complexity of the discussed problems for the three most commonly used types of interval linear programs is summarized in Table 2. For those problems, which remain (co-)NP-hard for programs with a fixed matrix, we have strengthened the previously known results by formulating new proofs that also hold for the considered special case.

**Acknowledgements** The first two authors were supported by the Czech Science Foundation under the project P402/13-10660S and by the Charles University, project GA UK No. 156317. The work of the first author was also supported by the grant SVV-2017-260452. The work of the third author was supported by the Czech Science Foundation Project no. 17-13086S.

## References

1. Ben-Tal, A., Ghaoui, L., Nemirovski, A.: Robust Optimization. In: Princeton Series in Applied Mathematics. Princeton University Press (2009)

2. Garajová, E.: The optimal solution set of interval linear programming problems. Master's thesis, Charles University, Prague (2016). <http://is.cuni.cz/webapps/zzp/detail/168259/?lang=en>
3. Gerlach, W.: Zur Lösung linearer Ungleichungssysteme bei Störung der rechten Seite und der Koeffizientenmatrix. *Math. Operationsforsch. Stat. Ser. Optim.* **12**(1), 41–43 (1981)
4. Hladík, M.: Complexity of necessary efficiency in interval linear programming and multiobjective linear programming. *Optim. Lett.* **6**(5), 893–899 (2012)
5. Hladík, M.: Interval linear programming: a survey. In: Mann, Z.A. (ed.) *Linear Programming—New Frontiers in Theory and Applications*, Chap. 2, pp. 85–120. Nova Science Publishers, New York (2012)
6. Hladík, M.: Weak and strong solvability of interval linear systems of equations and inequalities. *Linear Algebra Appl.* **438**(11), 4156–4165 (2013). <https://doi.org/10.1016/j.laa.2013.02.012>
7. Koníčková, J.: Strong unboundedness of interval linear programming problems. In: 12th GAMM—IMACS International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics (SCAN 2006), p. 26 (2006)
8. Li, W., Luo, J., Wang, Q., Li, Y.: Checking weak optimality of the solution to linear programming with interval right-hand side. *Optim. Lett.* **8**(4), 1287–1299 (2014)
9. Rohn, J.: Interval linear programming. In: *Linear Optimization Problems with Inexact Data*, pp. 79–100. Springer, US (2006)
10. Rohn, J.: Solvability of systems of interval linear equations and inequalities. In: *Linear Optimization Problems with Inexact Data*, pp. 35–77. Springer, US (2006)

# Bounding Multistage Stochastic Programs: A Scenario Tree Based Approach

Francesca Maggioni and Elisabetta Allevi

**Abstract** Multistage mixed-integer stochastic programs are among the most challenging optimization problems combining stochastic programs and discrete optimization problems. Approximation techniques which provide lower and upper bounds to the optimal value are very useful in practice. In this paper we present a critic summary of the results in Maggioni et al., *J Optim Theory Appl* 163:200–229 (2014), [4] and in Maggioni et al., *Comput Manag Sci* 13:423–457 (2016), [5] where we consider bounds based on the assumption that a sufficiently large discretized scenario tree describing the problem uncertainty is given but is unsolvable. Bounds based on group subproblems, quality of the deterministic solution and rolling-horizon approximation will be then discussed and compared.

**Keywords** Multistage stochastic programs · Bounds · Group subproblems

## 1 Introduction

In general the uncertainty of multistage stochastic programs is defined by means of a scenario process which may take uncountable infinite values. In order to solve it, is possible to consider a sufficiently large discretized scenario tree describing the uncertainty and considering it as benchmark. However in most of the real cases this problem is unsolvable requiring the inclusion of a large number of samples. Bounding its solution is then of practical interests.

---

*The authors would like to dedicate this work to the memory of Marida Bertocchi, exceptional colleague and friend sadly passed away on November 16, 2016.*

---

F. Maggioni (✉)

University Of Bergamo, Via dei Caniana 2, 24127 Bergamo, Italy  
e-mail: francesca.maggioni@unibg.it

E. Allevi

University Of Brescia, Via S. Chiara 50, 25122 Brescia, Italy  
e-mail: allevi@unibs.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_41

The aim of the paper is to present a brief and critic summary of bounds in multi-stage mixed-integer stochastic programs introduced in [4, 5] based on the assumption that a sufficiently large discretized scenario tree describing the problem uncertainty is given but is unsolvable. Chain of lower bounds less complex than the original problem are solved by solving sets of group subproblems made by fixed and free scenarios, and taking an expectation across scenario groups. Monotonicity results are provided. Other approximations of the optimal stochastic solution have been quantified by the introduction of measures of the quality of the deterministic solution and rolling horizon measures which update the estimation and add more information at each stage. The general idea behind construction of the proposed bounds, is that for every optimization problem of minimization type, lower bounds to the optimal solution can be found by relaxation of constraints and upper bound by inserting feasible solutions. Bounds for multistage convex problems with concave risk functionals based on scenario tree approaches are also provided in [7]. Other approaches bounding the infinite problem are presented in [8].

The paper is organized as follows: Sect. 2 introduces the notation and basic definitions. Lower bounds based on solving group subproblems are in Sect. 3 and upper bounds for the optimal multistage objective value are in Sect. 4. Section 5 concludes the paper.

## 2 Preliminaries

We consider the following scenario formulation of a *multistage mixed-integer stochastic program* (see [11]):

$$\begin{aligned}
 RP &= \min_x E_{\xi^{H-1}} z(x, \xi^{H-1}) \\
 &= \min_{x^1, \dots, x^H} c^1 x^1 + \sum_{s=1}^S \pi_s (c^2(\xi_s^1) x^2(\xi_s) + \dots + c^H(\xi_s^{H-1}) x^H(\xi_s)) \\
 &\text{s.t. } Ax^1 = h^1, \\
 &\quad T^1(\xi_s^1) x^1(\xi_s) + W^2(\xi_s^1) x^2(\xi_s) = h^2(\xi_s^1), \quad s = 1, \dots, S, \\
 &\quad \vdots \\
 &\quad T^{H-1}(\xi_s^{H-1}) x^{H-1}(\xi_s) + W^H(\xi_s^{H-1}) x^H(\xi_s) = h^H(\xi_s^{H-1}), \quad s = 1, \dots, S, \\
 &\quad x^t(\xi_{j'}^t) = x^t(\xi_{j''}^t), \forall j', j'' \in \{1, \dots, S\} \text{ for which } \xi_{j'}^t = \xi_{j''}^t, \quad t = 2, \dots, H,
 \end{aligned} \tag{1}$$

where  $c^1 \in \mathbb{R}^{n_1}$  and  $h^1 \in \mathbb{R}^{m_1}$  are known vectors,  $A \in \mathbb{R}^{m_1 \times n_1}$  is a known matrix,  $h^t \in \mathbb{R}^{m_t}$ ,  $c^t \in \mathbb{R}^{n_t}$ ,  $T^{t-1} \in \mathbb{R}^{m_t \times n_{t-1}}$ ,  $W^t \in \mathbb{R}^{m_t \times n_t}$ ,  $t = 2, \dots, H$  are the uncertain parameter vectors and matrices. The random process  $\xi^t$ ,  $t = 1, \dots, H - 1$ , is revealed gradually over time in  $H$  periods and  $\xi^t := (\xi^1, \dots, \xi^t)$ ,  $t = 1, \dots, H - 1$  denotes the history of the process up to time  $t$ .  $\xi^t$  is defined on a probability space  $(\Xi^t, \mathcal{A}^t, p)$  with support  $\Xi^t \in \mathbb{R}^{n_t}$  and given probability distribution  $p$  on the  $\sigma$ -algebra  $\mathcal{A}^t$  (with

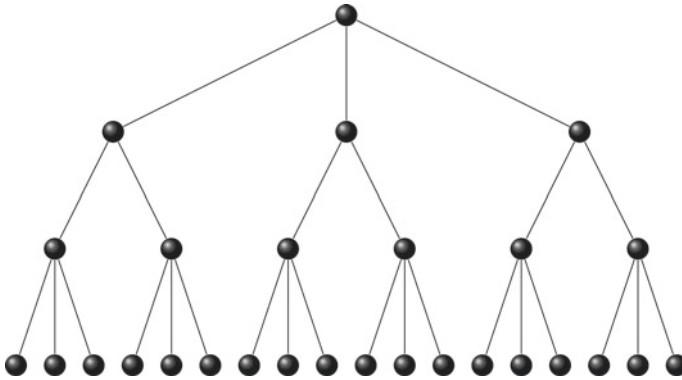


Fig. 1 Scenario tree representation of the random process  $\xi^{H-1}$  with approximate distribution

$\mathcal{A}^t \subseteq \mathcal{A}^{t+1}$ ) and  $E_{\xi^t}$  denotes the expectation with respect to  $\xi^t$ . Let  $\xi_1, \dots, \xi_S$ , be the possible realizations (or scenarios) of  $\xi^{H-1}$ ,  $\Xi$  the finite support of possible scenarios and  $\xi_s^t$  the history of the  $s$ -realization,  $s = 1, \dots, S$ , up to stage  $t$ ,  $t = 1, \dots, H - 1$ . Let  $\pi_s$  the probability of scenario  $s$ ,  $s = 1, \dots, S$ . See Fig. 1 for a scenario tree visualization of the scenario process with approximate distribution. The decision vector  $\mathbf{x} := (x^1, x^2, \dots, x^H)$ , with  $x^t \in \mathbb{R}_+^{n_t-d_t} \times \mathbb{N}^{d_t}$ ,  $t = 1, \dots, H$ , is *nonanticipative* which means it depends on the information up to time  $t$ . The last set of constraints enforce this condition. In the following, for a simpler presentation, the feasibility condition on  $x^t$  will be omitted even if assumed to be satisfied.

The main principle to obtain lower bounds of problem (1) is given by the relaxation of some constraints. This is the case of the *multistage wait-and-see* problem (WS), where the *nonanticipativity* constraints are relaxed. WS is then obtained by averaging the total costs of the  $S$  deterministic programs:

$$\begin{aligned}
 WS = & \sum_{s=1}^S \pi_s \min_{x^1(\xi_s), \dots, x^H(\xi_s)} c^1 x^1(\xi_s) + c^2(\xi_s^1) x^2(\xi_s) + \dots + c^H(\xi_s^{H-1}) x^H(\xi_s) \\
 \text{s.t. } & Ax^1(\xi_s) = h^1, \\
 & T^1(\xi_s^1) x^1(\xi_s) + W^2(\xi_s^1) x^2(\xi_s) = h^2(\xi_s^1), \\
 & \vdots \\
 & T^{H-1}(\xi_s^{H-1}) x^{H-1}(\xi_s) + W^H(\xi_s^{H-1}) x^H(\xi_s) = h^H(\xi_s^{H-1}).
 \end{aligned} \tag{2}$$

The *Expected Value problem EV* is obtained by replacing all random parameters by their expected values and solving the deterministic program, with  $\bar{\xi} := (\bar{\xi}^1, \bar{\xi}^2, \dots, \bar{\xi}^{H-1}) = (E\xi^1, E\xi^2, \dots, E\xi^{H-1})$ :

$$EV := \min_{\mathbf{x}} z(\mathbf{x}, \bar{\xi}). \tag{3}$$



### 2.1 Basic Bounds

The following relations between  $RP$ ,  $WS$  and  $EV$  have been proved in [3].

**Theorem 1** For multistage stochastic mixed-integer programs of the form (1), the following inequalities hold true

$$WS \leq RP \leq EEV, \tag{4}$$

where  $EEV$  denotes the solution value of the  $RP$  model, having the first stage decision variables fixed at the optimal values obtained by using the expected value of coefficients.

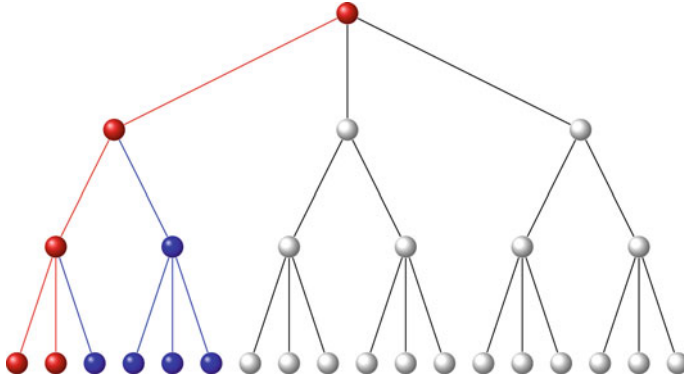
Similarly  $EEV^t$ ,  $t = 1, \dots, H - 1$  (see [2, 4]), is defined by fixing the decision variables up to stage  $t$  of  $RP$  at the optimal values obtained by using the problem  $EV$ . The Value of the Stochastic Solution at stage  $t$ ,  $VSS^t$  is then defined as  $VSS^t := EEV^t - RP$ ,  $t = 1, \dots, H - 1$ .

However, in several problems of practical interest the difference between  $EEV^t$  and  $WS$  is quite large. In the next sections we will discuss how to solve simpler problems for finding lower and upper bounds and proceed to find tighter and tighter bounds to  $RP$ .

### 3 Lower Bounds by Group Subproblems

In order to obtain lower bounds on  $RP$  problem which improve the left-hand side inequality in (4), one can solve smaller problems than the original one. The proposed approach (see [5]) divides a given problem into independent subproblems. We suppose to fix a number  $1 \leq R < S$  of reference scenarios among the possible  $S$  scenarios. Let  $\mathcal{R} = \{1, \dots, R\}$  be the index set of fixed scenarios. Without loss of generality we suppose they are the first  $R$  scenarios among the available  $S$  ones. We choose among the  $K = S - R$  scenarios ( $\xi_i$ ,  $i = R + 1, \dots, S$ ) a subgroup of cardinality  $k = 1, \dots, K$ . Let  $\mathcal{K} = \{R + 1, \dots, S\}$  be the index set of scenarios excluding those belonging to the fixed scenario set  $\mathcal{R}$ . Let  $\mathcal{P}(\mathcal{K})$  the power set of  $\mathcal{K}$  excluding the empty set. Let  $\mathcal{P}_k(\mathcal{K})$  the set of all subset of  $\mathcal{P}(\mathcal{K})$  with cardinality  $k$ . For any subset  $\Psi_k \in \mathcal{P}_k(\mathcal{K})$ , let  $\pi(\Psi_k) = \sum_{i \in \Psi_k} \pi_i$  be the probability assigned to scenarios group  $\Psi_k$ .

**Definition 1** For any given scenario group  $\Psi_k$ , the group subproblem  $MGR(\Psi_k, R)$  is defined as  $\min z^R(\Psi_k) :=$



**Fig. 2** Representation of the Group Subproblem  $MGR(\Psi_4, 2)$  with  $R = 2$  reference scenarios (in red) and one subset  $\Psi_4$  (in blue)

$$\begin{aligned}
 & \min_{x^1, \dots, x^H} \left( c^1 x^1 + \sum_{r=1}^R \left( \pi_r \sum_{t=2}^H c^t(\xi_r^{t-1}) x^t(\xi_r) \right) + \left( 1 - \sum_{r=1}^R \pi_r \right) \sum_{i \in \Psi_k} \frac{\pi_i}{\pi(\Psi_k)} \sum_{t=2}^H c^t(\xi_i^{t-1}) x^t(\xi_i) \right) \\
 & \text{s.t. } Ax^1 = h^1, \\
 & T^{t-1}(\xi_r^{t-1}) x^{t-1}(\xi_r) + W^t(\xi_r^{t-1}) x^t(\xi_r) = h^t(\xi_r^{t-1}), \quad r \in \mathcal{R}, \quad t = 2, \dots, H \quad (5) \\
 & T^{t-1}(\xi_i^{t-1}) x^{t-1}(\xi_i) + W^t(\xi_i^{t-1}) x^t(\xi_i) = h^t(\xi_i^{t-1}), \quad i \in \Psi_k, \quad t = 2, \dots, H \\
 & x^t(\xi_{j'}) = x^t(\xi_{j''}), \quad \forall j', j'' \in \mathcal{R} \cup \Psi_k \text{ for which } \xi_{j'}^t = \xi_{j''}^t, \quad t = 2, \dots, H.
 \end{aligned}$$

*Remark 1*  $MGR(\Psi_1, 1)$  reduces to the definition of *PAIRS* subproblem (see [4]).

A representation of  $MGR(\Psi_4, 2)$  is shown in Fig. 2 with  $R = 2$  reference scenarios (in red) and one subset  $\Psi_4$  (in blue).

**Definition 2** Given an integer  $k \in \{1, \dots, K\}$ , and  $R$  fixed scenarios, the *Multistage Expected value of the Group Subproblem Objective* function with  $k$  scenarios in each group and  $R$  fixed scenarios,  $MEGSO(k, R)$  is defined as

$$MEGSO(k, R) := \frac{1}{\binom{K-1}{k-1} \left( 1 - \sum_{r=1}^R \pi_r \right)} \left[ \sum_{\Psi_k \in \mathcal{P}_k(\mathcal{X})} \pi(\Psi_k) \min z^R(\Psi_k) \right]. \quad (6)$$

*Remark 2* The *Multistage Sum of Pairs Expected Values*, *MSPEV* [4] reduces to  $MEGSO(1, 1)$  as follows

$$MSPEV = MEGSO(1, 1) = \frac{1}{1 - \pi_a} \sum_{\Psi_1 \in \mathcal{P}_1(\mathcal{X})} \pi(\Psi_1) \min z^P(\Psi_1). \quad (7)$$

**Theorem 2** *Given an integer  $R$ ,  $1 \leq R < S$  and an integer  $k$ ,  $1 \leq k \leq K$  the following chains of inequalities hold true*

$$WS \leq \text{MEGSO}(1, R) \leq \text{MEGSO}(2, R) \leq \dots \leq \text{MEGSO}(K, R) = RP, \tag{8}$$

$$\text{MEGSO}(k, 1) \leq \text{MEGSO}(k, 2) \leq \dots \leq \text{MEGSO}(k, S - k) = RP. \tag{9}$$

Theorem 2 says that *MEGSO* is monotonically nondecreasing in the cardinality  $k$  of scenarios of the subsets  $\Psi_k$  with  $R$  fixed and monotonically nondecreasing in the number of reference scenarios  $R$  with  $k$  fixed. The proof can be found in [5].

## 4 Upper Bounds

In this section we discuss and compare some types of upper bounds for multistage mixed-integer programs. We focus on bounds based on solving group subproblems, quality measures of the deterministic solution and on rolling horizon measures.

### 4.1 Upper Bounds from Multistage Group Subproblems

In this section we recall upper bounds for multistage stochastic programs based on solving group subproblems (see [10] for the two-stage case and [5] for the multistage one). Given  $\check{\mathbf{x}}_R^1$  the optimal first stage solution of the stochastic problem  $\min_{\mathbf{x}} z(\mathbf{x}, \xi_1, \dots, \xi_R)$ , based only on the  $R$  reference scenarios, then a possible upper bound of  $RP$  is:

$$\text{MEVRS}^{1,R} := E_{\xi^{H-1}} \min_{\mathbf{x}^{(2,H)}} z(\check{\mathbf{x}}_R^1, \mathbf{x}^{(2,H)}, \xi^{H-1}). \tag{10}$$

A tighter upper bound to  $RP$  is the *Multistage Expectation of Group Subproblems*  $\text{MEGS}(k, R)$ , which represents the minimum optimal value among those obtained by solving the original stochastic program (1), using the optimal first stage solution  $\hat{\mathbf{x}}_{\Psi_k, R}^1$  of each group subproblem (5). This can be expressed as follows:

$$\text{MEGS}(k, R) := \min_{\Psi_k \in \mathcal{P}_k(\mathcal{K}) \cup \mathcal{R}} (E_{\xi^{H-1}} \min_{\mathbf{x}^{(2,H)}} z(\hat{\mathbf{x}}_{\Psi_k, R}^1, \mathbf{x}^{(2,H)}, \xi^{H-1})). \tag{11}$$

The following inequality holds (see [5]).

**Proposition 1** For a fixed number  $R$  of reference scenarios and any  $1 \leq k \leq K$  we have

$$RP \leq MEGS(k, R) \leq MEVRS^{1,R}. \tag{12}$$

### 4.2 Upper Bounds Based on the Expected Value Skeleton Solution

Measures of the structure (skeleton) of the deterministic solution such as the *Multi-stage Loss Using the Skeleton Solution MLUSS* has been introduced in [4], in relation to the standard *VSS* (see in [9] the definition in the two-stage case). The aim of these measures is to identify meaningful information, which can be extracted from the solution of the deterministic problem, in order to reduce the size of the stochastic one.  $MLUSS^t$  are computed as the difference between the optimal values of the stochastic problem  $RP$  and its reduced version  $MESSV^t$  obtained by fixing the out-of-basis variables up to stage in the expected value solution. Having a  $MLUSS^t$  close to zero suggests that the out-of-basis variables chosen by the expected value model until stage  $t$  are correct also in a stochastic environment. The following relations hold true (see [4]):

**Proposition 2**

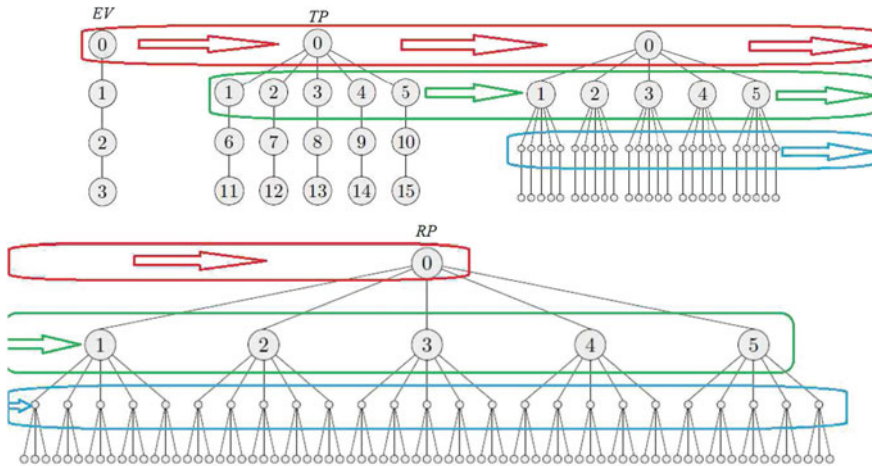
$$MLUSS^{t+1} \geq MLUSS^t, \quad t = 1, \dots, H - 2, \tag{13}$$

$$RP \leq MESSV^t \leq EEV^t, \quad t = 1, \dots, H - 1. \tag{14}$$

In these lines, other measures of the goodness of deterministic solutions based on reduced costs of the deterministic solution are proposed in [1].

### 4.3 Upper bounds based on Rolling Horizon approaches

Multistage problems such as  $EEV^t$  are often infeasible since they require to fix many variables to their value obtained via the expected value model. An alternative approach to consider is the *rolling time horizon* procedure taking into account the arrival of new information at each stage. This is obtained by solving a sequence of  $H$  scenario trees with random parameters in periods  $t, \dots, H - 1$  replaced with their expected value and solve the associated model with fixing the solutions obtained in the previous steps (see Fig. 3). The *Rolling Horizon Value of the Expected Value Solution* is then given by the difference with respect to  $RP$ . In similar way other rolling horizon measures are defined in [4] and in [6].



**Fig. 3** Procedure to compute the rolling horizon value of the expected value solution

## 5 Conclusions

In this paper lower and upper bounds for mixed-integer multistage stochastic programs have been discussed and compared. The bounds are based on the assumption that a sufficiently large scenario-tree process is given as approximation of the general infinite problem and it is considered as a benchmark. The lower and upper bounds proposed are based on groups subproblems, quality of deterministic solution and rolling horizon approaches. The approach discussed is both of theoretical and practical importance arising when solving problems of large instances where it is fundamental to have approximations of the optimal solution.

## References

1. Crainic, T.G., Maggioni, F., Perboli, G., Rei, W.: Reduced Cost-Based Variable Fixing in Two-Stage Stochastic Programming (2017) Available via CIRRELT Reports
2. Escudero, L.F., Garín, A., Merino, M., Pérez, G.: The value of the stochastic solution in multistage problems. *Top* **15**, 48–64 (2007)
3. Madansky, A.: Inequalities for stochastic linear programming problems. *Manag. Sci.* **6**, 197–204 (1960)
4. Maggioni, F., Allevi, E., Bertocchi, M.: Bounds in multistage linear stochastic programming. *J. Optim. Theory Appl.* **163**(1), 200–229 (2014)
5. Maggioni, F., Allevi, E., Bertocchi, M.: Monotonic bounds in multistage mixed-integer linear stochastic programming. *Comput. Manag. Sci.* **13**(3), 423–457 (2016)
6. Maggioni, F., Kaut, M., Bertazzi, L.: Stochastic Optimization models for a single-sink transportation problem. *Comput. Manag. Sci.* **6**, 251–267 (2009)
7. Maggioni, F., Pflug, G.C.: Bounds and approximations for multistage stochastic programs. *SIAM J. Optim.* **26**(1), 831–855 (2016)

8. Maggioni, F., Pflug, G.C.: Guaranteed Bounds for General Non-discrete Multistage Risk-Averse Stochastic Optimization Programs (2017) Submitted to SIOPT
9. Maggioni, F., Wallace, W.S.: Analyzing the quality of the expected value solution in stochastic programming. *Ann. Oper. Res.* **200**(1), 37–54 (2012)
10. Sandikçi, B., Kong, N., Schaefer, A.J.: A hierarchy of bounds for stochastic mixed-integer programs. *Math. Program. Ser. A* **138**(1), 253–272 (2012)
11. Shapiro, A., Dencheva, D., Ruszczyński, A.: *Lectures on Stochastic Programming: Modeling and Theory*. MPS-SIAM Series on Optimization (2009)

# A Polyhedral Study of the Robust Capacitated Edge Activation Problem

Sara Mattia

**Abstract** Given a capacitated network, the Capacitated Edge Activation problem consists of activating a minimum cost set of edges in order to serve some traffic demands. If the demands are subject to uncertainty, we speak of the Robust Capacitated Edge Activation problem. We consider the capacity formulation of the robust problem and study the corresponding polyhedron to generalize to the robust problem the results that are known for the problem without uncertainty.

**Keywords** Polyhedral study · Capacity formulation · Robustness

## 1 Introduction

Modern telecommunications networks must be resilient to traffic peaks and possible failures. Therefore, to increase the reliability of the system, they are designed to have redundant components. One common type of redundancy is to install extra-capacity on the edges. Keeping all the redundant components active is very expensive, both from an economical point of view and from a sustainability perspective. Then, those components are usually deactivated and switched on only when a fault occurs [1, 23]. The *Capacitated Edge Activation* (CEA) problem selects the minimum cost set of edges to be activated to guarantee the routing of a set of demands. Differently for the Network Loading (NL) problem, where the capacities must be computed and are not subject to restrictions, the edges in the CEA problem have given (bounded) capacities. When the demands are subject to uncertainty, it must be ensured that all the traffic matrices belonging to a given uncertainty set are served. This is the *Robust Capacitated Edge Activation* (RCEA) problem. Different versions of the problem can be defined according to the possibility to split the demands and to adapt the routing to the specific traffic matrix that is considered. If the commodities

---

S. Mattia (✉)

Istituto di Analisi dei Sistemi ed Informatica (IASI), Consiglio Nazionale delle Ricerche (CNR), via dei Taurini 19, 00185 Roma, Italy  
e-mail: sara.mattia@iasi.cnr.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217, DOI 10.1007/978-3-319-67308-0\_42

are restricted to follow a single path in the network, we speak of *unsplittable* flows (denoted by *unsplit*). Otherwise, we have *splittable* flows (denoted by *split*). If the routing must be the same, independently of the specific traffic matrix to be served, it is said to be *static* (denoted by *stat*). If it can be changed according to the matrix, it is called *dynamic* (denoted by *dyn*).

In this paper we consider a *capacity* formulation of the problem, that is, a formulation that contains only design variables (the ones related to the activation of the edges), but no routing variables (the ones that define the routes for the demands). Instead, we call *flow* formulation a model including both design and routing variables. For splittable flows, capacity formulations are usually obtained through Benders-like reformulations. Benders decomposition [6] is popular technique that has been applied to many problems, ranging from personnel scheduling [25] to black-out prevention [9]. For applications of this framework to network design problems see [12, 24]. For unsplittable flows instead, some kinds of combinatorial Benders cuts [10, 11, 17, 23] are usually needed. We study the polyhedron corresponding to the capacity formulation and generalize to the RCEA problem some results that are known for the problem without uncertain demands [17, 23], presenting results for splittable and unsplittable flows, static and dynamic routing. As far as we know, this is the first time that the polyhedral properties of the RCEA problem are investigated. In Sect. 2 we present a review of the literature. In Sect. 3 polyhedral results are derived. In Sect. 4 conclusions are discussed.

## 2 Literature

The NL problem is very popular in the literature: many approaches have been proposed and different formulations and the related polyhedra have been investigated. We address the reader to [4, 5, 8, 18, 26] and references therein for additional details. Uncertainty in the demands has also been considered: properties and algorithms for some robust versions of the RNL problem and related problems are investigated in [3, 15, 21, 22, 24], among the others. Lagrangian relaxations for the flow formulation of the splittable CEA problem are presented in [13]. A Lagrangian heuristic to be used within a branch-and-bound approach to solve the splittable CEA problem is illustrated [14]. The CEA problem with unsplittable flows and survivability requirements is considered in [1]. The authors present valid inequalities and derive a branch-and-cut algorithm to solve the problem, but no capacity formulation is used. A problem with unsplittable flows where multiple capacity modules can be separately activated on the edges of a network is studied in [7]. Valid inequalities for the polyhedron of the single edge problem to be added to a flow formulation are presented and a branch-and-cut algorithm is proposed.

Results for the capacity formulation of the CEA problem with splittable flows where a connected network is also required are presented in [2]. Valid inequalities and polyhedral results for the capacity formulation of the CEA problem with splittable and unsplittable flows are presented in [23]. The author shows that the *tight metric*



*inequalities*, that completely define the polyhedron of the capacity formulation of many network design problems [4, 19–21], do not provide the complete description of the polyhedron corresponding to the CEA problem. As far as we know, there is no paper in the literature presenting a polyhedral study of the capacity formulation of the RCEA problem.

### 3 Polyhedral Results

Denote by  $G(V, E)$  the undirected graph representing the network and by  $K$  the set of node pairs (commodities) to be served. For each  $k \in K$ ,  $s_k$  is the source,  $t_k$  the destination and  $d_k \geq 0$ , which is uncertain, the amount. Different amounts correspond to different traffic matrices. Let  $\mathcal{D}$  be a non-empty and bounded set including the traffic matrices that must be served and assume that there is at least one matrix in  $\mathcal{D}$  such that  $d_k > 0$  for any  $k \in K$ . Let  $c_e > 0$  be the cost of activating edge  $e$ . Assume that, when activated, all the edges provide the same capacity, that we denote by  $U$ . Let  $R = \{split, unsplit\}$  be the set of the routing policies and let  $F = \{stat, dyn\}$  be the set of the flows policies.

$T \subseteq E$  is a *bridge set* if the simultaneous removal from the network of the edges in  $T$  makes the problem infeasible. The status of  $T$  depends on the demands, on the flows and on the routing, as  $T$  may be a bridge set for a given demand/flow/routing triple  $(\mathcal{D}, f, r)$ , but not for another one. We say that a bridge set is minimal for  $(\mathcal{D}, f, r)$  if it does not strictly contain another bridge set. Denote by  $\mathcal{B}(\mathcal{D}, f, r)$  the minimal bridge sets for  $\mathcal{D}$  if flows  $f$  and routing  $r$  are used. Let  $x_e$  be a binary variable taking value one if  $e$  is active and zero otherwise. The RCEA problem can be formulated as below.

$$\begin{aligned}
 \text{SC}(\mathcal{D}, f, r) \quad & \min \sum_{e \in E} c_e x_e \\
 & \sum_{e \in B} x_e \geq 1 \quad B \in \mathcal{B}(\mathcal{D}, f, r) \\
 & \mathbf{x} \in \{0, 1\}^{|E|}
 \end{aligned}$$

Trivially,  $E$  is a bridge set. A *cut* is a bi-partition of  $V$  into  $\{S, V \setminus S\}$ . Given a cut, let  $\delta(S)$  and  $K(S)$  be edges having endpoints in different sets of the partition and the demands separated by the cut. If  $K(S) \neq \emptyset$ , then  $\delta(S)$  is a bridge set, see Example 1. A bridge set not corresponding to a cut is reported in Example 2.

*Example 1* Consider a complete graph on three nodes where  $U = 1$ . Let the uncertainty set be  $\mathcal{D} = \text{conv}\{D^1, D^2\}$ , where  $d_{12}^1 = 1, d_{ij}^1 = 0$  otherwise,  $d_{13}^2 = 1, d_{ij}^2 = 0$  otherwise. Consider cut  $\{\{1\}, \{2, 3\}\}$ , with  $\delta(S) = \{(1, 2), (1, 3)\}$ . Set  $\delta(S)$  defines a minimal bridge set for any  $r \in R, f \in F$ .

*Example 2* Consider a complete graph on four nodes. Assume that  $U = 1$ . Let  $\mathcal{D} = \text{conv}\{D^1, D^2\}$ , where  $d_{13}^1 = d_{14}^1 = 1, d_{ij}^1 = 0$  otherwise and  $d_{13}^2 = d_{23}^2 = 1, d_{ij}^2 = 0$  otherwise.  $T = \{(1, 3), (2, 4)\} \subseteq E$  is a minimal bridge set for  $(\mathcal{D}, unsplit, stat)$ .

Remove the edges in  $T$ . The only feasible unsplittable routing for  $D^1$  is  $d_{14}$  routed on  $\{(1, 4)\}$  and  $d_{13}$  routed on  $\{(1, 2), (2, 3)\}$ . It is not possible to route  $D^2$  if we keep the same routing used in  $D^1$  for  $d_{13}^2$ .

Note that the set  $T$  in Example 2 is not a bridge set for  $(\mathcal{D}, \text{split}, r)$ , for  $r \in R$  or for  $(\mathcal{D}, f, \text{dyn})$ , for  $f \in F$ . Moreover, it does not correspond to the edge set  $\delta(S)$  of any cut  $\{S, V \setminus S\}$ . Let  $P(\mathcal{D}, f, r)$  denote the convex hull of the integer feasible solutions of formulation  $\text{SC}(\mathcal{D}, f, r)$ . We assume that there is no bridge set of cardinality one, that is, the removal of a single edge from the network does not make the problem infeasible. We now generalize to the RCEA problem some results presented in [17, 23] for the CEA problem.

**Lemma 1**  $P(\mathcal{D}, f, r)$  is full-dimensional.

*Proof* Let  $\mathbf{x}^e$  be the solution having  $x_h^e = 1$  for  $h \in E \setminus \{e\}$  and  $x_e^e = 0$ . Since there are no single edge bridge sets,  $\mathbf{x}^e$  is feasible for any  $e$ . Let  $\mathbf{y}$  be the solution with  $y_h = 1$  for all  $h \in E$ . Vectors  $\mathbf{y}$  and  $\mathbf{x}^e$  for  $e \in E$  are  $|E| + 1$  affinely independent vectors in  $P(\mathcal{D}, f, r)$ .  $\square$

Simple valid inequalities are:

$$x_e \geq 0 \quad e \in E \tag{1}$$

$$x_e \leq 1 \quad e \in E \tag{2}$$

**Theorem 1** The following results hold:

1. inequalities (1) are facet-defining for  $P(\mathcal{D}, f, r)$  for  $r \in R$  and  $f \in F$ , if  $\{e, h\} \notin \mathcal{B}(\mathcal{D}, f, r)$ , for any  $h \in E \setminus \{e\}$ ;
2. inequalities (2) are facet-defining for  $P(\mathcal{D}, f, r)$  for  $r \in R$  and  $f \in F$ .

*Proof* Part 1. One can find  $|E|$  affinely independent vectors in  $P(\mathcal{D}, f, r)$  as follows. Let  $\mathbf{y}$  be the solution having  $y_q = 1$  for all  $q \in E \setminus \{e\}$  and  $y_e = 0$ . For any  $h \in E \setminus \{e\}$ , let  $\mathbf{x}^{eh}$  be the solution with  $x_q^{eh} = 0$  for  $q \in \{e, h\}$  and  $x_q^{eh} = 1$  otherwise. Part 2. For  $t \neq e$ , consider vector  $\mathbf{z}^{te}$  having  $z_q^{te} = 1$  for all  $q \in E \setminus \{t\}$  and  $z_t^{te} = 0$ . Vectors  $\mathbf{w} = \mathbf{1}$  and  $\mathbf{z}^{te}$  for  $t \in E \setminus \{e\}$  are  $|E|$  affinely independent vectors of  $P(\mathcal{D}, f, r)$ .  $\square$

No facet of the form  $\mathbf{a}^T \mathbf{x} \geq b$  can have negative coefficients, but for inequalities (2).

**Theorem 2** Let  $\mathbf{a}^T \mathbf{x} \geq b$  be a valid inequality for  $P(\mathcal{D}, f, r)$ . If  $\exists e : a_e < 0$ , then either  $\mathbf{a}^T \mathbf{x} \geq b$  is the upper bound inequality (2) for edge  $e$  or it is not a facet of  $P(\mathcal{D}, f, r)$ .

*Proof* Assume that  $\mathbf{a}^T \mathbf{x} \geq b$  is a facet and that there exists one edge  $e$  having  $a_e < 0$ . Suppose that  $\mathbf{y}$  is a feasible solution satisfying  $\mathbf{a}^T \mathbf{x} \geq b$  with equality and that  $y_e = 0$ . Then, solution  $\mathbf{z}$  obtained as  $z_t = y_t$  for  $t \in E \setminus \{e\}$ ,  $z_e = 1$  is still feasible, but  $\mathbf{a}^T \mathbf{z} = \mathbf{a}^T \mathbf{y} - a_e = b - a_e < b$ . Hence, any feasible solution that satisfies  $\mathbf{a}^T \mathbf{x} \geq b$  with equality also satisfies facet  $x_e \leq 1$  with equality.  $\square$

Given a cut, the *cutset* inequalities (3) are valid for  $P(\mathcal{D}, f, r)$ .

$$\sum_{e \in \delta(S)} x_e \geq \max_{K \in \mathcal{D}} \left\lceil \frac{\sum_{k \in K(S)} d_k}{U} \right\rceil \tag{3}$$

Indeed, if the active edges of a cut do not support the maximum demand separated by the cut, no feasible solution exists. Consider now a two-node problem, where the graph may have parallel edges and commodities. Let  $P_2(\mathcal{D}, f, r)$  be the corresponding polyhedron.

**Theorem 3** *Inequalities (3) are facet-defining for  $P_2(\mathcal{D}, \text{split}, r)$  for  $r \in R$ , if and only if  $|E| > \max_{K \in \mathcal{D}} \lceil \sum_{k \in K(S)} d_k / U \rceil > 0$ .*

*Proof* Since the flows are splittable and we are considering a two-node problem, the commodities in  $K$  can be replaced by a unique commodity with source node 1, destination node 2 and demand  $d_{12} = \max_{K \in \mathcal{D}} \sum_{k \in K(S)} d_k / U$ . Let  $D(S) = \lceil d_{12} \rceil$ . Part 1. Since the maximum flow across the cut can be as large as  $U|E|$ , if  $|E| < D(S)$  no integer feasible  $\mathbf{x}$  exists. Part 2. If  $D(S) = 0$ , then the cutset inequality is dominated by the non-negativity constraints. If  $|E| = D(S)$ , then each edge is a bridge and the cutset inequality is dominated by the upper bound constraints. Assume then that  $|E| > D(S)$ , hence no set of  $r = |E| - D(S)$  edges is a bridge set. Let  $\mathbf{y}^i$  be the vector having  $D(S)$  consecutive ones starting from entry  $e_i$  (restarting from  $e_1$  once the last edge is reached), whereas the other entries are equal to zero. Vectors  $\mathbf{y}^i$  for  $e_i \in E$  are  $|E|$  affinely independent feasible solutions satisfying the inequality with equality.  $\square$

Given a cut  $\{S, V \setminus S\}$ , a corresponding two-node problem can be derived by shrinking each subset into a node. We denote the polyhedron of the two-node problem associated with cut  $\{S, V \setminus S\}$  by  $P_2^S(\mathcal{D}, \text{split}, r)$ . It is possible to prove that, under some conditions, the cutset inequalities that are facets of  $P_2^S(\mathcal{D}, \text{split}, r)$ , are facets of  $P(\mathcal{D}, \text{split}, r)$  as well. Let  $R_2^S(\mathcal{D}, \text{split}, r) \subseteq P_2^S(\mathcal{D}, \text{split}, r)$  be the integer feasible solutions satisfying the cutset inequality with equality. Consider a problem where the activation status of the edges in  $J \subseteq E$  has already been decided. Let  $\mathcal{B}^J(\mathcal{D}, \text{split}, r, \bar{\mathbf{x}})$  denote the minimal bridge sets of the problem with demand set  $\mathcal{D}$ , flows  $f$ , routing  $r$  and activation status  $\bar{\mathbf{x}}$  for the edges in  $J$ .

**Theorem 4** *If  $S$  and  $V \subseteq S$  are connected,  $|E| > \lceil \max_{K \in \mathcal{D}} \sum_{k \in K(S)} d_k / U \rceil > 0$  and  $\{e\} \notin \mathcal{B}^{\delta(S)}(\mathcal{D}, \text{split}, r, \bar{\mathbf{x}})$  for any  $e \in E \setminus \delta(S)$ ,  $\bar{\mathbf{x}} \in R_2^S(\mathcal{D}, \text{split}, r)$ , then the corresponding cutset inequality is a facet of  $P(\mathcal{D}, \text{split}, r)$ .*

*Proof* Suppose that  $S$  is not connected and let  $V_1^1, \dots, V_i^q$  be its connected components. Then the inequality corresponding to  $\{S, V \setminus S\}$  is dominated by the ones corresponding to cuts  $\{V_i, V_i \setminus S\}$ ,  $i = 1, \dots, q$ . The same if  $V \setminus S$  is not connected. Assume that  $S$  and  $V \setminus S$  are connected. Since the conditions of Theorem 3 are met, then the inequality is a facet of  $P_2^S(\mathcal{D}, \text{split}, r)$ . Let  $\mathbf{z}^1, \dots, \mathbf{z}^{|\delta(S)|}$  be affinely independent vectors of  $R_2^S(\mathcal{D}, \text{split}, r)$ . Let  $\mathbf{y}^i \in \{0, 1\}^{|E|}$  be the vector having  $y_e^i = z_e^i$  for  $e \in \delta(S)$ ,  $y_e^i = 1$  for  $e \in E \setminus \delta(S)$ . Choose a vector  $\mathbf{z}^j$ . For any  $t \in E \setminus \delta(S)$  let  $\mathbf{w}^t$

be the vector having  $w_e^t = z_e^j$  for  $e \in \delta(S)$ ,  $w_e^t = 1$  for  $e \in E \setminus \delta(S) \setminus \{t\}$ ,  $w_e^t = 0$  for  $e = t$ . Vectors  $\mathbf{y}^i$  for  $i \in \delta(S)$  and  $\mathbf{w}^t$  for  $t \in E \setminus \delta(S)$  are  $|\delta(S)| + |E \setminus \delta(S)|$  affinely independent feasible solutions of  $P(\mathcal{D}, \text{split}, r)$  satisfying the inequality with equality.  $\square$

On the contrary, the cutset inequalities defined above are valid for the unsplittable RCEA problem, but they are not facets for  $P_2(\mathcal{D}, \text{unsplit}, r)$ , as the RCEA problem includes the CEA problem (and then the Bin-Packing problem) as special case [17, 23].

Exact separation of cutset inequalities is, in general, NP-hard, even for a problem without uncertainty, as it corresponds to solve a Max-Cut problem [5]. Cutset inequalities can be separated, for example, by adapting the heuristic algorithm used for the RNL problem in [21]. For unsplittable flows, the right-hand-side of the inequalities can be strengthened as in [17, 23] by applying the technique originally proposed for the Bin-Packing problem in [16].

## 4 Conclusions

We studied from the polyhedral point of view the capacity formulation of the robust version of the CEA problem. We identified facets and valid inequalities, providing the first polyhedral results for this problem. In fact, we generalized to the RCEA problem the results known for the CEA problem.

## References

1. Addis, B., Carello, G., Mattia, S.: Survivable green traffic engineering with shared protection. *Networks* **69**(1), 6–22 (2017)
2. Agarwal, Y., Aneja, Y.: Fixed charge multicommodity network design using p-partition facets. *Eur. J. Oper. Res.* **258**, 124–135 (2017)
3. Altın, A., Yaman, H., Pınar, M.: The robust network loading problem under hose demand uncertainty: formulation, polyhedral analysis, and computations. *INFORMS J. Comput.* **23**(1), 75–89 (2011)
4. Avella, P., Mattia, S., Sassano, A.: Metric inequalities and the network loading problem. *Discret. Optim.* **4**, 103–114 (2007)
5. Barahona, F.: Network design using cut inequalities. *SIAM J. Optim.* **6**, 823–834 (1996)
6. Benders, J.F.: Partitioning procedures for solving mixed-variables programming problems. *Numer. Math.* **4**, 238–252 (1962)
7. Benhamichea, A., Mahjoub, R., Perrot, N., Uchoa, E.: Unsplittable non-additive capacitated network design using set functions polyhedra. *Comput. Oper. Res.* **66**, 105–115 (2016)
8. Bienstock, D., Chopra, S., Günlük, O., Tsai, C.Y.: Minimum cost capacity installation for multicommodity network flows. *Math. Program. B* **81**, 177–199 (1998)
9. Bienstock, D., Mattia, S.: Using mixed-integer programming to solve power grid blackout problems. *Discret. Optim.* **4**, 115–141 (2007)
10. Botton, Q., Fortz, B., Gouveia, L., Poss, M.: Benders decomposition for the hop-constrained survivable network design problem. *INFORMS J. Comput.* **25**, 13–26 (2013)

11. Codato, G., Fischetti, M.: Combinatorial Benders' cuts for mixed-integer linear programming. *Oper. Res.* **54**(4), 756–766 (2006)
12. Costa, A.: A survey on benders decomposition applied to fixed-charge network design problems. *Comput. Oper. Res.* **32**, 1429–1450 (2005)
13. Crainic, T., Frangioni, A., Gendron, B.: Bundle-based relaxation methods for multicommodity capacitated fixed charge network design. *Discret. Appl. Math.* **112**, 73–99 (2001)
14. Holmberg, K., Yuan, D.: A lagrangian heuristic based branch-and-bound approach for the capacitated network design problem. *Oper. Res.* **48**, 461–481 (2000)
15. Lee, C., Lee, K., Park, S.: Benders decomposition approach for the robust network design problem with flow bifurcations. *Networks* **62**(1), 1–16 (2013)
16. Martello, S., Toth, P.: Lower bounds and reduction procedures for the bin packing problem. *Disc. Appl. Math.* **28**, 59–70 (1990)
17. Mattia, S.: The capacity formulation of the capacitated edge activation problem. Submitted
18. Mattia, S.: Separating tight metric inequalities by bilevel programming. *Oper. Res. Lett.* **40**(6), 568–572 (2012)
19. Mattia, S.: Solving survivable two-layer network design problems by metric inequalities. *Comput. Optim. Appl.* **51**(2), 809–834 (2012)
20. Mattia, S.: A polyhedral study of the capacity formulation of the multilayer network design problem. *Networks* **62**(1), 17–26 (2013)
21. Mattia, S.: The robust network loading problem with dynamic routing. *Comput. Optim. Appl.* **54**(3), 619–643 (2013)
22. Mattia, S.: The cut property under demand uncertainty. *Networks* **66**(2), 159–168 (2015)
23. Mattia, S.: Benders decomposition for capacitated network design. In: *Proceedings of ISCO 2016*. LNCS, vol. 9849, pp. 71–80 (2016)
24. Mattia, S., Poss, M.: A comparison of different routing schemes for the robust network loading problem: polyhedral results and computation. Submitted
25. Mattia, S., Rossi, F., Servilio, M., Smriglio, S.: Staffing and scheduling flexible call centers by two-stage robust optimization. *Omega* **72**, 25–37 (2017)
26. Raack, C., Koster, A., Orlowski, S., Wessälly, R.: On cut-based inequalities for capacitated network design polyhedra. *Networks* **57**(2), 141–156 (2011)

# Performance Evaluation of a Push Merge System with Multiple Suppliers, an Intermediate Buffer and a Distribution Center with Parallel Channels: The Erlang Case

Despoina Ntio, Michael Vidalis, Stelios Koukoumialos  
and Alexandros Diamantidis

**Abstract** In this paper, the most important performance measures of a two stage, push merge system are estimated. More specifically,  $S$  non identical and reliable suppliers send material to a distribution center (DC) of one product type. The DC has  $N$  possible parallel identical reliable machines. Between the suppliers and the DC a buffer with unlimited capacity, is located in order the material flow to be controlled. All stations processing and replenishment times are assumed stochastic and follow the Erlang distribution. The considered model is developed as Markov Process with discrete states where the Matrix Analytic method is used to calculate the system stationary probabilities and performance measures.

**Keywords** Merge production systems • Markovian analysis • Matrix analytic method

---

D. Ntio (✉)  
Technical Educational Institute of Kozani, Kozani, Greece  
e-mail: dntio@teikoz.gr

M. Vidalis  
Department of Business Administration,  
University of the Aegean, Chios Island, Greece  
e-mail: mvid@ba.aegean.gr

S. Koukoumialos  
Department of Business Administration,  
Technical Educational Institute of Larissa, Larissa, Greece  
e-mail: skoukoum@teilar.gr

A. Diamantidis  
Department of Economics, Aristotle University  
of Thessaloniki, Thessaloniki, Greece  
e-mail: adiama@econ.auth.gr

# 1 Introduction and Literature Review

Production systems are networks that consist of work stations, distribution centers and buffers. Production systems structure must serve the purpose and the needs of production procedure. According to the production type, materials or finished goods pass through the system in a specific way. Also, every station is located so as the material flow to be facilitated. As a result, a production system topology take various forms such as serial, merge or convergent, split or divergent or general topology. Both the production systems structure and complexity have crucial impact on its performance measures. The most important performance measures of production systems are the throughput (average production rate of the system), the work in process (average inventory in the system or in the buffer), the Cycle Time or Flow Time (the average time that an item remains in the system). This work focuses on the performance measure computation of a two stage, push merge system.

Merge production systems are extension of the classical production lines. They are more complex networks where many upstream machines converge to only one downstream node. Queuing theory has a great contribution on the study of production systems. In the existing literature there are many researchers that adopt the queuing theory framework for the analysis of production systems. For example, [1], analyzed a production system with one station, [2], proposed a method that decomposed an initial system into individual and less complex queues. In [3], the authors examined production systems with mixed topology (merge and split). [4], studied a production system that was capable of producing many type of products. The model examined in this work is also developed using the framework of queuing theory.

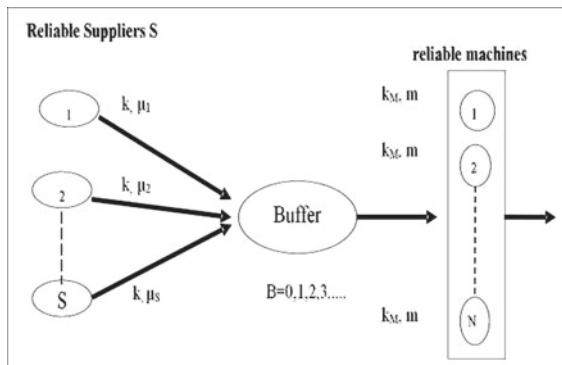
Moreover, in the literature many studies examine systems with unreliable machines. More Specifically, [5–7], studied two stage unreliable merge production systems with continuous material flow. [8, 9], analyzed a two stage unreliable merge system with two suppliers, a shared buffer and one downstream machine. [10], examined a production system with multiple product types using the decomposition approach. Finally, at the field of systems with unreliable machines [11], examined a two stage system with two parallel machines at the first stage, an intermediate buffer, and two parallel machines at the second stage. Considering merge systems with reliable machines, most studies examine discrete material flow systems with two stages, and stochastic processing times. Most of the models were developed as Markovian processes with discrete states. More specifically, [12], studied a two stage system with two suppliers, a buffer and one machine at DC where all machines processing times followed the Erlang distribution. [13, 14], examined push merge systems with multiple reliable merging suppliers an intermediate buffer and a downstream DC with multiple reliable machines. [15] examined a push-pull merge system with multiple merging suppliers where the demand was covered from the finished goods buffer. [16] examined a push-pull merge-split system with two merging suppliers where the demand was satisfied by

two retailers. This study is a clear extension of the work presented in [12, 13] where the exponential distribution was used to model all processing and replenishment times. The model considered here differs significantly from the models of those two studies. More specifically, in this work the Erlang distribution is adopted for modeling the processing times which enables the study of systems with different variability.

## 2 Description of the Model

This study deals with a two echelon merge production network of a single product. The examined system is a push system with an infinite number of merging suppliers  $S$ , an intermediate shared buffer with infinite capacity, a DC with an infinite number of parallel machines and infinite number of Erlang phases for all processing times (Fig. 1). Infinite number has the meaning that the proposed algorithm can handle theoretically (depending always on the computational limits of the hardware that executes the algorithm) any large value for these four parameters. The merging suppliers are reliable and non identical and the parallel machines at the DC, are also reliable and identical without failures. Moreover, there is an unlimited supply of materials towards the suppliers and an unlimited capacity shipping area after the distribution center. Furthermore, the processing times of the merging suppliers and of the parallel machines at the DC are stochastic and follow the Erlang phase type distribution. All merging suppliers have the same number of Erlang phases and all parallel machines at the DC have the same number of Erlang phases. The processing goods, due to Erlang distribution, must definitely pass through  $K$  number of phases. The physical interpretation of Erlang phases could be the production of sub processes such as construction, assembly, packaging, distribution e.t.c. The Erlang distribution is used because it has smaller coefficient of variation (CV) than the exponential distribution, in order to capture real life situations. The practical added value is the exploration of how these phases of the Erlang distribution can affect the

**Fig. 1** A merge production system with infinite, suppliers, buffer capacity and parallel machines at the DC





performance measures of the supply chain. Finally, blocking appears (at the suppliers) when one or more suppliers have finished their process, the buffer is full and all parallel machines at the DC are occupied. We should note that there is no blocking at the DC, due to the unlimited capacity shipping area after the specific center.

The blocking remains until at least one of the machines at the DC finishes processing. As a result, one unit from the buffer enters the DC and the system becomes unblocked. In case where more than one supplier is blocked, the supplier with the smallest index has priority over the others to send its finished product to the buffer. Suppliers are ranked in ascending order indicating their unblocking priority. For the model variable, the following notation is introduced:

- $S$ , the number of merging suppliers of the system,  $S = 1, 2, 3, 4, 5, \dots$ , (infinite)
- $B$ , the capacity of buffer,  $B = 1, 2, 3, 4, 5, \dots$ , (infinite)
- $N$ , the number of machines in the DC,  $N = 1, 2, 3, 4, 5, \dots$ , (infinite)
- $\mu = (\mu_1, \mu_2, \dots, \mu_S)$  the vector of mean processing rate for every supplier  $i = 1, 2, \dots, S$
- $m = (m, m, \dots, m)$  the vector of mean processing rate for every  $j = 1, 2, \dots, N$  machine of DC.
- $k = 1, 2, 3, 4, 5, \dots$ , the number of Erlang phases for every  $i = 1, 2, \dots, S$  supplier.
- $k_M = 1, 2, 3, 4, 5, \dots$ , the number of Erlang phases for every  $j = 1, 2, \dots, N$  machine at the DC.

### 3 Solution Procedure and Structure of the Transition Matrix

The above production system is analyzed as a continuous time Markov process with a finite number of states. If we want to determine the stationary probabilities, a set of linear flow balance equations need to be solved. Each flow equation is associated with one state of the system. If we have systems with a large number of states, and we want to find exact solutions, the only way to do this is if *structural properties of the equations or equivalently of the transition matrix structure can be exploited* [17].

The structure of the transition matrix is affected by the following system parameters: number of suppliers ( $S$ ), capacity of distribution centre (DC), capacity of buffer ( $B$ ) and number of Erlang phases for every  $i = 1, 2, \dots, S$  supplier and every  $j = 1, 2, \dots, N$  machine of the DC. Since the exanimate system can be viewed as a birth-death stochastic process (births are the inputs to the buffer and DC and deaths are the outputs from the DC) the transition matrix is a diagonal matrix which consists in general of three group of sub-matrices: The sub-matrices in the main diagonal, the sub-matrices above the main diagonal and the sub-matrices below the

main diagonal. The existence of the Erlang distribution and the fact that the Erlang phases affect directly the structure of the transition matrix makes the considered system much more complicated than a similar system in which the exponential distribution could have been used instead. Furthermore the way that the transition matrix is generated is also more complicated in comparison with the exponential case.

Specifically, in order to create the transition matrix  $P_{(k, k_M)}^{S, B, N}$  of the whole system we must follow a reductive procedure contained five steps: (i) creation of the initiate matrix (generator matrix) of the simplest system  $P_{(2, 2)}^{1, 0, 1}$ , (ii) creation of the transition matrix  $P_{(k, k_M)}^{1, 0, 1}$  according to the previous generator matrix and to the influence of Erlang phases  $k$  and  $k_m$ , (iii) structure of the matrix  $P_{(k, k_M)}^{1, B, 1}$  with the influence of buffer capacity  $B$  using steps (ii), (iv) using the matrix in previous step we create the matrix  $P_{(k, k_M)}^{S, B, 1}$  taking into consideration the influence of the number of suppliers of the system  $S$ , and finally (v) creation of the final transition matrix  $P_{(k, k_M)}^{S, B, N}$  taking into consideration the number of machines in the D.C.,  $N$ , the number of Erlang phases  $k$ ,  $k_M$ , and the transition matrix  $P_{(k, k_M)}^{S, B, 1}$ , from the previous step.

#### 4 Performance Measures and Validation of the Model

Once the steady-state probabilities have been calculated, all the performance measures of the system can be estimated. The most important performance measures of the system are the average number of work in process of the system ( $WIP_{system}$ ), the average number of work in process of the buffer ( $WIP_{buffer}$ ), the average number of work in process of the D.C. ( $WIP_{D.C.}$ ), the average number of work in process of the final process ( $WIP_{finalprocess}$ ), the mean output rate or throughput of the system ( $THR$ ), and the mean time that a unit remains in the system ( $CT$ ). More specifically:

$WIP_{system}$  = number of suppliers  $S$  + mean buffer level + Mean number of Occupied machines in DC,  $WIP_{buffer}$  = mean buffer level + Mean number of Blocked Parallel Servers,  $WIP_{D.C.}$  = Mean number of Occupied machines in DC,

$$WIP_{finalprocess} = WIP_{buffer} + WIP_{D.C}$$

$$THR = (\text{prob machines of D.C. to be to the last phase } k_m \text{ of the process}) * k_m * m,$$

$$CT = WIP_{system} / THR.$$

All the above performance measures, which were taken from the analytical Markovian model, were validated against simulation. For all the examined cases,

the percentage error between the simulation results and the results obtained from the algorithm for all the performance measures is negligible, and fluctuates between 0.02 and 0.20%.

## 5 Numerical Results

This section presents the impact of each parameter variability on the performance measures of the considered system. For this purpose extensive numerical results were obtained assuming that only one system parameter changes at a time (while others remain constant). Due to space limitation the tables with the numerical results are omitted but are available on request. Therefore only the conclusions obtained by the analysis of the numerical results are presented. The main tool used to describe the impact of each parameter variability is the elasticity of the performance measures as function of the system parameters. The numerical results obtained for balanced systems (systems where mean processing rate for all merging suppliers = mean processing rate for all machines at the DC) for the case where all the suppliers were identical.

Considering the throughput ( $THR$ ) as the explored performance measure, the numerical results showed that a reduction of the coefficient of variation ( $CV$ ) for the suppliers and the DC processing times leads to an increase of the throughput. The  $CV$  elasticity of the throughput was found to be  $-0.2124$  meaning that reducing  $CV$  by 1% has as a result an increment of throughput by 0.2124%. The impact of the buffer size on the throughput was also measured. The main conclusion is that an increment of the buffer size leads to an increment of the throughput and the buffer elasticity of throughput was found to be 0.0245. Another interesting conclusion is that the above elasticity value decreases when the buffer size and the number of suppliers increase simultaneously.

Furthermore the impact of the number of suppliers on the throughput was also measured. The numerical results showed that an increase of the number of the suppliers leads to an increase of the throughput and the number of suppliers elasticity of the throughput was also found to be equal to 0.0099.

Comparing all the above mentioned values of elasticity the main conclusion is that the reduction of the  $CV$  processing times of the suppliers and the DC, have the highest impact on the throughput increment, followed by the buffer size increment, whereas the increment of the number of suppliers have the lowest impact on the throughput increment.

When the examined performance measure is  $WIP_{system}$ , the numerical results showed that a reduction of the coefficient of variation ( $CV$ ) for the suppliers and the DC processing times leads to an increase of the  $WIP_{system}$ . The  $CV$  elasticity of the  $WIP_{system}$  was found to be  $-0.0601$ . The impact of the buffer size on the  $WIP_{system}$  was also measured. The main conclusion is that an increment of the buffer size leads to an increment of the  $WIP_{system}$  and the buffer elasticity of  $WIP_{system}$  was found to

be 0.1576. Also the above elasticity value decreases when the buffer size and the number of suppliers increase simultaneously.

Furthermore the impact of the number of suppliers on the  $WIP_{system}$  was also explored. The numerical results showed that an increase of the number of the suppliers leads to an increase of the  $WIP_{system}$  and the number of suppliers elasticity of the  $WIP_{system}$  was also found to be equal to 0.5961. Comparing all the above mentioned values of elasticity that main conclusion is that the increase of the number of suppliers have the highest impact on the  $WIP_{system}$  increment, followed by the buffer size increment, whereas the reduction of the  $CV$  processing times of the suppliers and the DC have the lowest impact on the  $WIP_{system}$  increment.

When the examined performance measure is  $WIP_{buffer}$ , the numerical results showed that a reduction of the coefficient of variation ( $CV$ ) for the suppliers and the DC processing times leads to a reduction of the  $WIP_{buffer}$ . The  $CV$  elasticity of the  $WIP_{buffer}$  was found to be 0.3686. Furthermore the impact of the number of suppliers on  $WIP_{buffer}$  was also measured. The numerical results showed that an increase of the number of the suppliers leads to an increase of the  $WIP_{buffer}$  and the number of suppliers elasticity of the  $WIP_{buffer}$  was found to be equal to 0.2243.

Apart from balanced systems, the behavior of unbalanced systems and especially systems under a constrained process (systems where mean processing rate for all merging suppliers  $>$  mean processing rate for all machines at the DC) was also explored. For unbalanced systems the main concern was to explore the evolution of all the performance measures ( $THR$ ,  $WIP_{system}$ ,  $WIP_{buffer}$ ,  $WIP_{D.C.}$  and  $CT$ ) as a function of specific system parameters such as  $S = 3, 4, \dots, 8$ ,  $B = 0, 2, 4, 6, 8, 10$ ,  $N = 1$ ,  $\mu_i = 1$ ,  $m_1 = 2$ , and  $k = k_m = 2$ . Due to space limitation only the evolution of the  $THR$  and the  $CT$  is presented in Figs.2a and 2b, respectively.  $THR$  initially increases while the number  $S$  increases but up to a specific number of  $S$  (Fig. 2a). From that point of  $S$ ,  $THR$  remain constant. The same happens with the influence of buffer capacity. There is a number of  $B$  that makes  $THR$  to remain again constant.

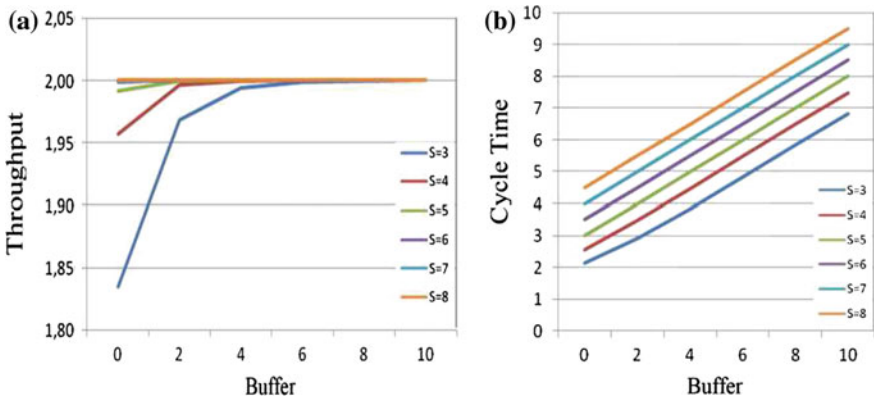


Fig. 2 Evolution of  $THR$  (a) and  $CT$  (b) for a constrained process system with  $S = 3, 4, \dots, 8$ ,  $B = 0, 2, 4, 6, 8, 10$ ,  $N = 1$ ,  $\mu_i = 1$ ,  $m_1 = 2$ , and  $k = k_m = 2$

Finally,  $CT$  increases while both number of  $S$  and capacity of  $B$  is increased (Fig. 2b). It is interesting that  $CT$  increases linearly as a function of  $B$ . A linear regression of  $CT$  based on  $B$  and  $S$  is a target of future research.

## 6 Further Research

Possible area for further research could be the study of similar systems with both unreliable suppliers and non identical unreliable machines at the DC, as well as systems with parallel servers at each merging supplier. Similarly, unbalanced systems should be analyzed thoroughly (using as an indicator elasticity) in order to explore the impact of the system parameters on the performance measures of such systems. Additionally, the considered model in this manuscript could be examined from an economic point of view to explore the optimal combination of the system parameters in order to minimize a cost function. Furthermore, an interesting topic could be the analysis of a similar push-pull system with external demand.

## References

1. Yao, D., Buzacott, A.: Queueing models for flexible machining station, part II: the method of coxian phases. *Eur. J. Oper. Res.* **39**, 241–252 (1985)
2. Lee, H.-S., Pollock, S.: Approximate analysis for the merge configuration of an open queueing network with blocking. *IIE Trans.* **21**, 122–124 (1989)
3. Kavusturucu, A., Gupta, S.M.: Manufacturing systems with machine vacations arbitrary topology and finite buffers. *Int. J. Prod. Econ.* **58**, 1–15 (1999)
4. Feng, W., Zheng, L., Li, J.: Scheduling policies in multi-product manufacturing systems with sequence-dependent setup times and finite buffers. *Int. J. Prod. Res.* **50**, 7479–7492 (2012)
5. Tan, B.: A three-station merge unreliable stations and a shared buffer. *Math. Comput. Model.* **33**, 1011–1026 (2001)
6. Helber, S., Mehrtens, N.: Exact analysis of a continuous material merge system with limited buffer capacity and three stations, In: Gershwin, S.B., Dallery, Papadopoulos, Y.C., T. Smith, J.M.: (eds.). *To Analysis and Modeling of Manufacturing Systems*. pp. 85–119, Springer (2003)
7. Helber, S., Jusic, H.: A new decomposition approach for non-cyclic continuous material flow lines with a merging flow of material. *Ann. Oper. Res.* **125**, 117–139 (2004)
8. Diamantidis, A.C., Papadopoulos, C.T., Vidalis, M.I.: Exact analysis of a discrete material three-station one-buffer merge system with unreliable machines. *Int. J. Prod. Res.* **42**, 651–675 (2004)
9. Diamantidis, A., Papadopoulos, C.: Markovian analysis of a discrete material manufacturing system with merge operations, operation-dependent and idleness failures. *Comput. Ind. Eng.* **50**, 466–487 (2006)
10. Colledani, M., Gandola, F., Matta, A., Tolio, T.: Performance evaluation of linear and non-linear multi-product multi-stage lines with unreliable machines and finite homogeneous buffers. *IIE Trans.* **40**, 612–626 (2008)
11. Liu, Y., Li, J.: Split and merge production systems: performance analysis and structural properties. *IIE Trans.* **42**, 422–434 (2010)

12. Vidalis, M., Koukoumialos, S., Ntio, D., Varlas, G.: Performance evaluation of a merge supply system with a distribution centre, two reliable suppliers, one buffer and Erlang lead time. *Int. J. Bus. Sci. Appl. Manag.* **7**(3), 42–55 (2012)
13. Vidalis, M.I., Koukoumialos, S., Geranios, M.: Performance evaluation of a merge supply network: A distribution centre with multiple reliable random suppliers. *Comput. Ind. Eng.* **70**, 43–58 (2014)
14. Vidalis, M., Koukoumialos, S., Diamantidis, A., Blanas, G.: Performance evaluation of a merge supply network: a production centre with multiple different reliable suppliers. In: 10th Conference on Stochastic Models of Manufacturing and Service Operations, Volos, pp. 255–262. (University Of Thessaly, Greece (2015)
15. Diamantidis, A., Koukoumialos, S., Vidalis, M.: Performance evaluation of a push-pull merge system with multiple suppliers, an intermediate buffer and a distribution center with parallel machines/channels. *Int. J. Prod. Res.* **54**(9), 2628–2652 (2016)
16. Diamantidis, A., Koukoumialos, S., Vidalis, M.: Markovian analysis of a push-pull merge system with two suppliers, an intermediate buffer and two retailers. *Int. J. Oper. Res. Inf. Syst.* **8**(2), 1–35 (2016)
17. Nelson, R.: Matrix geometric solution in markov models, a mathematical tutorial, IBM Research Division (1991)

# A Push Shipping-Dispatching Approach for High-Value Items: From Modeling to Managerial Insights

Jean Respen and Nicolas Zufferey

**Abstract** A real shipping-dispatching problem is considered in a three-level supply chain (plant, wholesalers, shops). Along the way, different perturbations are expected (when manufacturing, when forecasting the demand, and when dispatching the inventory from the wholesalers level), and accurate reactions must be taken. An integer linear program is proposed and some managerial insights are given.

**Keywords** Inventory management • Shipping • Dispatching • Simulation

## 1 Introduction

A company XYZ (it cannot be named because of a non-disclosure agreement) is facing a three-level inventory deployment problem (P). The considered supply chain is composed of a single factory (where high-value items of different products are manufactured), the wholesalers and the shops. There is no inventory policy from a downstream level to any upstream level: a push policy from the plant is imposed by XYZ, and any shortage at the shop level results in a lost sale. A shop can only be delivered from its single associated wholesaler. From a XYZ perspective, the shops are the final points of interest, and the final client, which actually buys an item, is not integrated in the supply chain as the shops are often not owned by XYZ.

The decisions are made at the plant level. Three perturbations are encountered: at the production-plan level (because of unreliable suppliers); at the dispatching level (because of unexpected shipping decisions made by the wholesalers); on the demand (as the actual demand is different from the forecasted demand). The planning horizon is a year. At the end of each time period (a week), the only decision to make is the

---

J. Respen  
Dootix Sàrl, 1630 Bulle, Switzerland  
e-mail: jean@dootix.com

N. Zufferey (✉)  
GSEM - University of Geneva, Geneva, Switzerland  
e-mail: n.zufferey@unige.ch

number of items of each product to send from the plant to each shop. A solution is a shipping plan for a whole year. The objective function involves three components: shortages, rarity (as opposed to the assortment diversity) and inventory. Because of the uncertainties, simulation is a relevant way to evaluate a solution.

Different surveys on simulation can be found in [3, 7, 17]. Simulation-optimization approaches are proposed in [14, 15] for inventory-management problems. When transportation costs are encountered, a multi-level inventory management problem is investigated in [9]. Centralized and decentralized ordering models are compared numerically. A two-level inventory problem is studied in [5], where order-up-to- $S$  replenishment policies (i.e., anytime an order is placed, it should bring the available inventory up to the level  $S$ ) are applied in the case retailers face different compound Poisson demand processes. A related problem is proposed in [1], where simple recursive procedures are developed for measuring the shortage costs. In [2], simulation is used to analyze distribution systems with stochastic demands. Advantages of echelon stock [4] and installation stock [8, 11] are evaluated. In [10], a survey is proposed on facility location and supply chain management, in addition to recent advances in optimization techniques for such problems and inventory deployment.

Relying on [13], where problem (P) and associated solution methods were introduced, the contributions of this work are the following. First, an integer linear program is proposed. The resulting exact method is able to tackle some markets faced by XYZ. Second, managerial insights are provided. The problem features are presented in Sect. 2. An ILP (integer linear programming) formulation is depicted in Sect. 3, along with a summary of the best-existing solution method. Experiments are conducted in Sect. 4. Section 5 concludes the paper.

## 2 Presentation of the Perturbations and of Problem (P)

For various reasons (e.g., suppliers being unreliable, problems in raw-material deliveries), the production plan can weekly suffer two different types of perturbation: (1) some products are not produced at all; (2) some items of some products are not produced, because of a lower production rate. Let  $\hat{p}_t^i$  be the planned number of items of product  $i$  to be produced in week  $t$ , and  $p_t^i$  be the corresponding actual number involving the perturbation. Each week, at most  $\delta_1\%$  of products are not produced, due to non deliveries from the supplier level. Thus,  $p_t^i = 0$  for these products, instead of  $p_t^i = \hat{p}_t^i$ . Two weeks are impacted by such a perturbation, and the missed production must be compensated later. In this case, we have  $p_t^i = p_{t+1}^i = 0$  for such products. XYZ imposed that  $p_{t+2}^i = \hat{p}_{t+2}^i + \hat{p}_t^i + \hat{p}_{t+1}^i$  and proposes  $\delta_1 \in [5\%, 15\%]$ . On the remaining  $1 - \delta_1$  percent of the products (which are produced), each product has a risk of  $\delta_2\%$  to be slowed down. It means that at most  $\delta_3\%$  of  $\hat{p}_t^i$  is produced in week  $t + 1$  instead of week  $t$ . At the end of week  $t$ , the total number of unproduced items because of a reduced production performance is  $Q_t = \hat{P}_t - P_t$ , where  $\hat{P}_t = \sum_i \hat{p}_t^i$  and  $P_t = \sum_i p_t^i$ .



On the total number  $\hat{P}_t$  of items expected to be produced at week  $t$ , a given threshold of  $\varepsilon\%$  of items are allowed to be slowed down, meaning that if  $Q_t \leq \varepsilon \cdot \hat{P}_t$ ,  $XYZ$  does not react. But if  $Q_t > \varepsilon \cdot \hat{P}_t$ ,  $C_t = Q_t - \varepsilon \cdot \hat{P}_t$  items have to be compensated (i.e., other products are manufactured instead of the planned ones). Products which are not slowed down at week  $t$  and planned at week  $t + 1$  are eligible for compensation, as well as the products which were not expected to be produced at week  $t$  (i.e.,  $\hat{p}_t^i = 0$ ) but are expected to be produced at week  $t + 1$  (i.e.,  $\hat{p}_{t+1}^i > 0$ ). For such eligible products, some items (randomly selected, one at a time) have to be produced in week  $t$ , as long as quantity  $C_t$  is not reached.  $XYZ$  proposes  $\delta_2 \in [45\%, 55\%]$ ,  $\delta_3 \in [5\%, 15\%]$ ,  $\varepsilon \in [5\%, 15\%]$ .

Unsurprisingly, there is a gap between the actual and the forecasted demands for the shop level. Based on the forecasted demand patterns given by  $XYZ$ , for each week and each product, a normal mean and a standard deviation of the perturbation is associated with each shop. The forecasted demand and the perturbation are thus used to generate the actual demand. The wholesalers belong to  $XYZ$ . It is expected that the different wholesalers ship the received items to the decided shops (at the plant level), but perturbations however occur (as each wholesaler has its favorite shops). With each shop  $j$  is associated a priority  $w_j$ , and let  $w^{max}$  (set to 3 by  $XYZ$ ) be the largest possible priority (i.e., the priority of the most important shops). Each wholesaler can modify the decisions (planned at the plant level) by sending the items to a shop  $j'$  that differs from the expected shop  $j$ .  $XYZ$  assumes that  $\delta_4\%$  of the items do not go to the expected shop. On these  $\delta_4\%$ ,  $\delta_5\%$  are shipped to a lower-priority shop.  $XYZ$  proposes  $\delta_4 \in [30\%, 50\%]$  and  $\delta_5 \in [60\%, 80\%]$ .

Problem (P) can be described as follows.  $N$  different products are manufactured in the plant.  $M$  is the total number of wholesalers. There are  $W$  weeks in a year (set to 47 by  $XYZ$ ). At the end of each week  $t$ ,  $p_t^i$  items of product  $i$  are shipped from the plant. As imposed by  $XYZ$ , all the items are weekly pushed (no inventory is kept in the plant). A set  $\mathcal{J}$  of shops must be supplied by the wholesalers level, and each shop  $j$  is associated with exactly one wholesaler  $m_j$ . In other words, we can have  $m_j = m_{j'}$  even if  $j \neq j'$ . For week  $t$ ,  $I_t^{i,m_j}$  denotes the on-hand inventory of wholesaler  $m_j$  associated with product  $i$ . Each wholesaler delivers to its associated shops and tries to have zero stock. Let  $\hat{x}_t^{i,j}$  be the number of items of product  $i$  supposed to arrive at shop  $j$  at week  $t$  from the corresponding wholesaler  $m_j$ .  $\hat{x}_t^{i,j}$  is a decision variable determined at the plant level. As each decision variable  $\hat{x}_t^{i,j}$  is an expected/planned value, let  $x_t^{i,j}$  denote the corresponding observed/simulated value. Moreover, there is a lead time of  $L_{m_j}$  weeks from the plant to the wholesaler  $m_j$ . Another lead time of  $L_j^{m_j}$  weeks occurs then to reach the shop  $j$  from the corresponding wholesaler  $m_j$ . For each week  $t$ , a demand  $\hat{d}_t^{i,j}$  is forecasted for product  $i$  at shop  $j$ , and let  $d_t^{i,j}$  be the corresponding actual demand. When the items  $x_t^{i,j}$  arrive in the shop  $j$ , the associated on-hand inventory of shop  $j$  is updated as follows:  $I_t^{i,j} = \max(0, I_{t-1}^{i,j} - d_{t-1}^{i,j} + x_t^{i,j})$ . We similarly have:  $\hat{I}_t^{i,j} = \max(0, \hat{I}_{t-1}^{i,j} - \hat{d}_{t-1}^{i,j} + \hat{x}_t^{i,j})$ . The inventory at period  $t$  is thus set as the inventory at period  $t - 1$  decreased by the demand at period  $t - 1$  and augmented by the number of delivered items. A solution  $\hat{S}$  is formulated as  $\hat{S} = (\hat{S}_1, \hat{S}_2, \dots, \hat{S}_t, \dots, \hat{S}_W)$ , which is

a per week list of  $\hat{S}_t$ 's, where  $\hat{S}_t = (\hat{S}_t^1, \hat{S}_t^2, \dots, \hat{S}_t^i, \dots, \hat{S}_t^N)$  is the corresponding solution for week  $t$  for each product  $i$ .  $\hat{S}_t^i = (\hat{x}_t^{i,1}, \hat{x}_t^{i,2}, \dots, \hat{x}_t^{i,j}, \dots, \hat{x}_t^{i,J})$  is a vector of decision variables for each shop  $j \in \mathcal{J}$ , each week  $t$  and each product  $i$ .

The considered objective function  $f$  is made of three components  $(f_1, f_2, f_3)$  that have to be minimized lexicographically (no deterioration on  $f_i$  can be compensated by improvements on  $f_{i+1}$ ).  $f_1$  is the expected shortage penalty of item  $i$  at week  $t$  in shop  $j$ . It occurs if  $\hat{I}_t^{i,j} < \hat{d}_t^{i,j}$ . Let  $\hat{B}_t^{i,j} = \max(0, \hat{d}_t^{i,j} - \hat{I}_t^{i,j})$  be the expected shortage quantity of product  $i$  at week  $t$  in shop  $j$ , and  $\hat{y}_t^{i,j}$  be a binary value which is 0 if  $\hat{x}_t^{i,j} > 0$ , and 1 otherwise.  $f_1$  is formulated in Eq. (1) as the sum of the shortage penalties, where a shortage that has started a long time ago costs more than a recent one. For any product  $i$  in shortage, XYZ assumes that if an item of product  $i$  arrives in a shop  $j$  at week  $t$ , then the corresponding shortage penalty is set back to 0 from that time period, even if the resulting on-hand inventory is below the demand.

$$f_1 = \sum_{j=1}^J w_j \sum_{t=1}^W \sum_{t'=1}^{t-1} (t - t') \sum_{i=1}^N \hat{B}_{t'}^{i,j} \cdot \hat{y}_t^{i,j} \text{ (shortage penalty)} \tag{1}$$

Let  $\hat{y}_t^{i,j} = 1$  if  $\hat{I}_t^{i,j} = 0$ , and  $\hat{y}_t^{i,j} = 0$  otherwise. Equation (2) presents  $f_2$  as a measure of rarity of each product  $i$  in each shop  $j$  for each week  $t$ .

$$f_2 = \frac{1}{J \cdot W \cdot N} \cdot \sum_{t=1}^W \sum_{j=1}^J \sum_{i=1}^N \hat{y}_t^{i,j} \text{ (rarity, as opposed to assortment diversity)} \tag{2}$$

Equation (3) formulates  $f_3$  as a weighted inventory penalty (the inventory cost is smaller for the shops with higher priorities, as the margin are larger for these shops).

$$f_3 = \sum_{j=1}^J [(w^{max} + 1) - w_j] \left[ \sum_{t=1}^W \sum_{i=1}^N \hat{I}_t^{i,j} \right] \text{ (inventory penalty)} \tag{3}$$

### 3 Exact and Solution Methods

Two types of methods are presented to generate solutions for (P): *EM* (an exact method using CPLEX) and *HM* (the best heuristic method). A simulator is needed to evaluate the actual value of any given planned solution (built with an optimization method). At the end, the decision-maker of XYZ can modify and evaluate again the solution, based on her/his expertise and on the non-modeled features.

The reader is referred to [13] for an accurate presentation of *HM*, which is somewhat related to ingredients from ant algorithms [22]. It consists of three steps: (1) generate a set  $A$  of  $a$  (parameter tuned to 100) solutions with the constructive heuristic *BUILD*; (2) construct a set  $B$  with the best (according to the expected value  $f$ )

$b$  (parameter tuned to 25) solutions of  $A$ ; (3) simulate the  $k$  (parameter tuned to 5) most distant solutions of  $B$  (as detailed in [13], these  $k$  solutions are quickly found with an ILP formulation using CPLEX, and a distance function able to measure the structural difference between any pair of solutions) and return the solution with the best simulated value. With such parameters, a single simulation roughly requires three minutes on the used computer, and  $HM$  needs always between 20 and 30 min to provide its final solution with the above three steps (as such computing times are in line with the needs of  $XYZ$ , we will not comment further on this aspect). Note that in each step of  $BUILD$ , one item is assigned to a randomly selected shop in the set of the most promising shops, which is computed based on the priority and on the shortage potential of each shop. As in [6],  $BUILD$  is based on a technique to accelerate the search by significantly reducing the decision space.

$f_1$  is not linear as it contains the multiplication of two decision variables. In order to use CPLEX and to allow benchmarking the results of  $HM$ , a linear model is proposed for (P). For this purpose,  $f_1$  is linearized as proposed in Eq. (4). The only difference is that  $f'_1$  does not reset the penalty to 0 if at least one item arrives at week  $t$ . Therefore, it is a reasonable approximation of  $f_1$  (i.e., a solution minimizing  $f'_1$  is likely to have many common features with a solution minimizing  $f_1$ ). Even more, it opens new perspectives for  $XYZ$  to measure the shortage penalties.

$$f'_1 = \sum_{j=1}^J w_j \sum_{t=1}^W \sum_{t'=1}^{t-1} (t - t') \sum_{i=1}^N \hat{B}_t^{i,j} \tag{4}$$

It is of course possible to keep a lexicographic approach while using a single objective function  $f$  with three components  $(f_1, f_2, f_3)$ . Indeed, let  $\alpha, \beta$  and  $\gamma$  be three coefficients such that  $\alpha = 1,000,000, \beta = 1,000$  and  $\gamma = 1$ . The objective function can then be formulated in Eq. (5). The three coefficient are different enough to preserve the lexicographic approach. The lexicographic multi-objective optimization approach proposed here is common practice, as reported in [12, 16, 18, 20, 21]. The constraints are given in Eqs. (6)–(10). Constraints (6) ensure a correct shortage computation by setting  $\hat{B}_t^{i,j}$  above  $\hat{d}_t^{i,j} - \hat{I}_t^{i,j}$ . Thanks to the minimization of  $f$ , this replaces the formulation of  $\hat{B}_t^{i,j} = \max(0, \hat{d}_t^{i,j} - \hat{I}_t^{i,j})$ . Constraints (7) and (8) ensure a correct computation of the inventory, which is updated using  $\hat{d}_{t-1}^{i,j}$  and  $\hat{x}_t^{i,j}$ , and is used to set  $\hat{y}_t^{i,j}$  correctly ( $\hat{y}_t^{i,j} = 1$  if  $\hat{I}_t^{i,j} = 0$ , and using the minimization of  $f$ ,  $\hat{y}_t^{i,j} = 0$  otherwise). Constraints (9) ensure that non-existent items are not created. Finally, Constraints (10) ensure that  $\hat{x}_t^{i,j}, \hat{B}_t^{i,j}$  and  $\hat{I}_t^{i,j}$  are non-negative integers. Below,  $t, i, j$  denote a time period, a product and a shop, respectively.

$$\min f = \alpha \cdot f_1 + \beta \cdot f_2 + \gamma \cdot f_3 \tag{5}$$

$$\hat{B}_t^{i,j} \geq \hat{d}_t^{i,j} - \hat{I}_t^{i,j}, \quad \forall t, i, j \tag{6}$$

$$\hat{I}_t^{i,j} \geq 1 - \hat{y}_t^{i,j}, \quad \forall t, i, j \tag{7}$$

$$\hat{I}_t^{i,j} \geq \hat{I}_{t-1}^{i,j} - \hat{d}_{t-1}^{i,j} + \hat{x}_t^{i,j}, \quad \forall t, i, j \tag{8}$$

$$\sum_j \hat{x}_t^{ij} \leq \hat{p}_{t-L_{m_j}-L_j}^i + \sum_j \hat{I}_{t-L_j}^{i,m_j}, \quad \forall t, i \tag{9}$$

$$\hat{x}_t^{ij}, \hat{B}_t^{ij}, \hat{I}_t^{ij} \in \mathbb{N}, \quad \forall t, i, j \tag{10}$$

## 4 Results

Tests were performed on an Intel Quad-core i7 @ 3.4 GHz with 8 GB DDR3 of RAM memory. An instance is composed of the following data: the involved market (defined by the number  $N$  of products, the number  $M$  of wholesalers, and the number  $J$  of shops with their associated priorities), the uncertainty parameters (i.e.,  $(\delta_1, \delta_2, \delta_3, \delta_4, \delta_5, \epsilon)$ ), and the perturbations of the demands, the production plan, the lead times, and the inventory levels (for wholesalers and shops). Based on the input from XYZ, the following reference values are used for the perturbations:  $\delta_1 = 10\%$ ,  $\delta_2 = 50\%$ ,  $\delta_3 = 10\%$ ,  $\delta_4 = 40\%$ ,  $\delta_5 = 70\%$ ,  $\epsilon = 10\%$ .

*EM* (tested with CPLEX 12.5) is compared with *HM* (which also uses  $f_1'$  as the first objective) for 25 instances, ranging from small to larger markets. CPLEX is tested with a time limit of 3 hours per objective and a memory limit of 7 GB (it was never reached). The three objectives are solved sequentially, which is a common practice. The model is first solved by ignoring  $f_2$  and  $f_3$ . Next,  $f_1^{(opt)}$  is set as a constraint and the model is solved minimizing  $f_2$  only. Finally,  $f_1^{(opt)}$  and  $f_2^{(opt)}$  are set as constraints, and the model is solved minimizing  $f_3$  only. The results are summarized in Table 1. The first three columns indicate the considered market  $(N, M, J)$ . The next three columns gives the output of *EM* for each objective-function component, where *O* stands for optimality (i.e., CPLEX finds the optimal solution), and *B* indicates that only an upper bound on the optimal value was returned within the allocated time limit. Anytime *B* is indicated for an objective, a “-” is given for the lower-level objectives. From a market size of  $(N, M, J) = (60, 7, 80)$ , no optimal information can be provided by *EM*. The last three columns give the percentage gaps of *HM* with respect to the results returned by *EM*. We can see that *EM* is able to find optimal solutions for a market size up to  $(N, M, J) = (24, 4, 30)$ . The results also allows benchmarking the efficiency of *HM*: it was never outperformed by *EM*, and it even finds better upper bounds (i.e., negative gaps) for the larger instances. Such negative gaps allow determining the market limitation of *EM*, which is typically  $(N, M, J) = (50, 6, 70)$ . Note that if the involved market is the whole world,  $(N, M, J)$  are roughly around  $(150, 10, 200)$ . In other words, even if *EM* could initially appear as very limited (in terms of the market size), it can however involve a couple of countries. When considering computing-time aspects, note that *HM* is also very efficient. For instance, *HM* requires only 580 s for the instance with  $(N, M, J) = (50, 6, 70)$ .

Managerial insights are now provided, with the goal to be as general as possible for the involved decision-maker *DM* (i.e., no specific market is considered).

**Table 1** Comparison of the exact method with *HM*

Instance			<i>EM</i> : output			<i>HM</i> : gap		
<i>N</i>	<i>M</i>	<i>J</i>	$f_1$	$f_2$	$f_3$	$f_1$ (%)	$f_2$ (%)	$f_3$ (%)
12	2	15	O	O	O	0.00	0.00	0.00
12	2	15	O	O	O	0.00	0.00	0.00
12	2	15	O	O	O	0.00	0.00	0.00
12	2	15	O	O	O	0.00	0.00	0.00
12	2	15	O	O	O	0.00	0.00	0.00
24	4	30	O	O	O	0.00	0.00	0.00
24	4	30	O	O	O	0.00	0.00	0.00
24	4	30	O	O	O	0.00	0.00	0.00
24	4	30	O	O	O	0.00	0.00	0.00
24	4	30	O	O	O	0.00	0.00	0.00
30	4	50	O	B	-	0.00	0.00	-
30	4	50	O	B	-	0.00	0.00	-
30	4	50	O	B	-	0.00	0.00	-
30	4	50	O	B	-	0.00	-0.05	-
30	4	50	O	B	-	0.00	0.02	-
50	6	70	B	-	-	0.00	-	-
50	6	70	B	-	-	0.00	-	-
50	6	70	B	-	-	-0.03	-	-
50	6	70	B	-	-	-0.01	-	-
50	6	70	B	-	-	0.00	-	-

*HM* is designed such that the best simulated solution (among  $k = 5$  satisfying solutions according to  $f$ ) is returned. Sometimes, due to non-modeled information, *DM* could decide to select another solution. For this purpose, we propose to return the  $k$  best solutions (provided with *HM*) to *DM*, thus *s/he* can select her/his preferred one among  $k$  options. It is thus important to propose five solutions with various structures (which is the case because of the distance function used within *HM*). In addition, we propose a measure *rob*, which is a reliability indicator of a simulation, defined as the standard deviation of  $f$  over the *sim* runs (knowing that a single simulation-run is based on the average results over  $sim = 40$  replications to be reliable). In realistic conditions, the lower  $rob(\hat{S})$  is, the higher is the probability that, if  $\hat{S}$  is used for multiple years, the corresponding actual solutions (with perturbations) are similar to each others. Moreover, it results in a good control on the costs. The above features are illustrated in Table 2, for which  $k = 5$  solutions (denoted as  $S_1$  to  $S_5$ ) are returned for a specific instance. For each solution and each objective-function component, its gap with respect to the best generated value is given. Solution  $S_3$  has the best  $f_1$ -value (as the indicated gap is 0), but it has a rather high *rob*-value. *DM* could decide to sacrifice 0.25% on  $f_1$  and select  $S_4$ , which has the lowest  $f_3$ -value and a

**Table 2** Comparison of five solutions provided by *HM*

Objective	$S_1$ (%)	$S_2$ (%)	$S_3$ (%)	$S_4$ (%)	$S_5$ (%)
$f_1$	1.31	0.05	0.00	0.25	0.87
$f_2$	0.04	0.00	0.12	0.01	0.18
$f_3$	1.58	0.63	0.75	0.00	1.25
<i>rob</i>	11.08	0.00	16.22	9.34	18.33

much better *rob*-value. Alternatively, s/he could select  $S_2$  by sacrificing 0.05% on  $f_1$  and get the best proposed values of  $f_2$  and *rob*. This example shows that *DM* could decide to sacrifice a small gap for one top objective and favor a lower-level objective because of the reliability indicator. Depending on the situation, *DM* could also decide to select the solution with the lowest *rob*-value, even if it is outperformed on the other objectives by the other solutions. In any case, the provided five solutions are competitive, as they are the most promising among  $a = 100$  solutions.

## 5 Conclusion

In this paper, we study a shipping-dispatching problem (P). We propose an integer linear program that is relevant for some markets encountered by the involved company XYZ. The exact model also allows showing the complexity of (P). We also provide managerial insights that can help decision-makers selecting a solution based on criteria that differs from quality only. Future works include the consideration of additional types of perturbations (e.g., on the lead-times). Moreover, XYZ could evaluate and optimize an integrated supply chain where suppliers, production and dispatching are centralized in a common system. It would allow XYZ reducing the perturbation parameters by reinforcing its control on the whole supply chain. A production planning allowing rejection and tardiness of jobs [19] would certainly inspire relevant models for an integrated approach.

## References

1. Axsäter, S.: Optimization of order-up-to-S policies in two-echelon inventory systems with periodic review. *Naval Res. Logistics* **40**(2), 245–253 (1993)
2. Axsäter, S., Juntti, L.: Comparison of echelon stock and installation stock policies for two-level inventory systems. *Int. J. Prod. Econ.* **45**(1), 303–310 (1996)
3. Banks, J.: *Handbook of Simulation*. Wiley Online Library (1998)
4. Clark, A.J., Scarf, H.: Optimal policies for a multi-echelon inventory problem. *Manage. Sci.* **6**(4), 475–490 (1960)
5. Forsberg, R.: Optimization of order-up-to-S policies for two-level inventory systems with compound Poisson demand. *Eur. J. Oper. Res.* **81**(1), 143–153 (1995)

6. Hertz, A., Schindl, D., Zufferey, N.: Lower bounding and tabu search procedures for the frequency assignment problem with polarization constraints. *4OR*, **3**(2), 139–161 (2005)
7. Law, A.M., Kelton, W.D.: *Simulation Modeling and Analysis*, vol. 2. McGraw-Hill, NY (1991)
8. Lee, H.L., Moinzadeh, K.: Two-parameter approximations for multi-echelon repairable inventory models with batch ordering policy. *IIE Trans.* **19**(2), 140–149 (1987)
9. Madadi, A., Kurz, M.E., Ashayeri, J.: Multi-level inventory management decisions with transportation cost consideration. *Transp. Res. Part E: Logistics Transp. Rev.* **46**(5), 719–734 (2010)
10. Melo, M.T., Nickel, S., Saldanha-da-Gama, F.: Facility location and supply chain management - A review. *Eur. J. Oper. Res.* **196**(2), 401–412 (2009)
11. Moinzadeh, K., Lee, H.L.: Batch size and stocking levels in multi-echelon repairable systems. *Manage. Sci.* **32**(12), 1567–1581 (1986)
12. Prats, X., Puig, V., Quevedo, J., Nejari, F.: Lexicographic optimisation for optimal departure aircraft trajectories. *Aerosp. Sci. Technol.* **14**(1), 26–37 (2010)
13. Respen, J., Zufferey, N., Wieser, Ph: Three-level inventory deployment for a luxury watch company facing various perturbations. *J. Oper. Res. Soc.* **68**(10), 1195–1210 (2017)
14. Silver, E.A., Zufferey, N.: Inventory control of raw materials under stochastic and seasonal lead times. *Int. J. Prod. Res.* **43**, 5161–5179 (2005)
15. Silver, E.A., Zufferey, N.: Inventory control of an item with a probabilistic replenishment lead time and a known supplier shutdown period. *Int. J. Prod. Res.* **49**(4), 923–947 (2011)
16. Solnon, C., Cung, V.D., Nguyen, A., Artigues, C.: The car sequencing problem: Overview of state-of-the-art methods and industrial case-study of the ROADEF 2005 challenge problem. *Eur. J. Oper. Res.* **191**(3), 912–927 (2008)
17. Terzi, S., Cavalieri, S.: Simulation in the supply chain context: a survey. *Comput. Ind.* **53**(1), 3–16 (2004)
18. Thevenin, S., Zufferey, N., Potvin, J.-Y.: Graph multi-coloring for a job scheduling application. *Discrete Appl. Math.* <https://doi.org/10.1016/j.dam.2016.05.023>
19. Thevenin, S., Zufferey, N., Widmer, M.: Metaheuristics for a scheduling problem with rejection and tardiness penalties. *J. Sched.* **18**(1), 89–105 (2015)
20. Thevenin, S., Zufferey, N., Widmer, M.: Order acceptance and scheduling with earliness and tardiness penalties. *J. Heuristics* **22**(6), 849–890 (2016)
21. Zufferey, N.: Heuristiques pour les Problèmes de la Coloration des Sommets d'un Graphe et d'Affectation de Fréquences avec Polarités. PhD thesis, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland (2002)
22. Zufferey, N.: Optimization by ant algorithms: possible roles for an individual ant. *Optim. Lett.* **6**(5), 963–973 (2012)

**Part IX**  
**Packing and Cutting**



# Optimization Models for Cut Sequencing

Claudio Arbib, Pasquale Avella, Maurizio Boccia, Fabrizio Marinelli  
and Sara Mattia

**Abstract** The paper describes models for scheduling the patterns that form a solution of a cutting stock problem. We highlight the problem of providing the required final products with the necessary items obtained from the cut, choosing which pattern feeds which lot of parts. This problem can be solved prior to schedule cuts, or in an integrated way. We present integer programming models for both approaches.

**Keywords** Cutting stock · Pattern sequencing · Integer programming

## 1 Introduction and Background

The well-known CUTTING STOCK PROBLEM (CSP) [7] calls for finding the best way of cutting a given set of small objects (*items* or parts) from larger ones of given shape. A *part type* indicates a class of items with identical features (shape, material etc.). A solution of a CSP consists of a partition of the item set into subsets  $K_1, \dots, K_n$  called *cuts*. A feasible cut has the geometric feature to admit a non-overlapping packing

---

C. Arbib (✉)

Università degli Studi dell'Aquila, L'Aquila, Italy

e-mail: claudio.arbib@univaq.it

P. Avella · M. Boccia

Università del Sannio, Benevento, Italy

e-mail: avella@unisannio.it

M. Boccia

e-mail: maurizio.boccia@unisannio.it

F. Marinelli

Università Politecnica delle Marche, Ancona, Italy

e-mail: fabrizio.marinelli@univpm.it

S. Mattia

IASI-CNR, Roma, Italy

e-mail: sara.mattia@iasi.cnr.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,

DOI 10.1007/978-3-319-67308-0\_45

of the items it consists of into a single large object. Identical cuts (that is, identical subsets  $K_i$ ) can be grouped into a single class called *pattern*, along with a description of the way in which the large object is to be cut. The number of cuts that belong to a pattern gives its *run length* and specifies how many times the cut must be replicated. Depending on application and technology, a cut can be operated on a single sheet or a sheet pack. In general, however, all the cuts of a pattern are operated consecutively to reduce slitter set-ups. So we will use the term “pattern” to indicate an indivisible cut operation that produces a known set of items.

The natural goal of a CSP is trim-loss minimization. In industrial production, however, not only one expects that cuts reduce material usage, but also that production is scheduled according to internal plant logistics and to order relevance/urgency. Therefore, a problem arises to decide how the cuts must be scheduled. When due-dates are associated with orders, a typical goal is the minimization of a due-date related objective function, such as the weighted sum of the orders tardiness [1], the maximum lateness of an order [2], or the weighted number of tardy jobs [3]. In other relevant cases, the objective is not related to order urgency but to production organization (e.g., minimizing the number of orders that are pending at any time [4, 5]) or general cost reduction (that may, e.g., include inventory [6]).

The scheduling problem can either be integrated with the CSP (as in [1, 2, 4–6]), or solved after an optimal CSP solution has been found [3, 8], i.e. first a CSP solution is found, then the resulting cuts are scheduled according to need: scheduling the cuts of a CSP solution to minimize the deviation of order completion times from due-dates is generally referred to as the PATTERN SEQUENCING PROBLEM (PSP).

The PSP can in turn be approached as a whole or by decomposition in two sub-problems:

- (i) assign items to lots corresponding to the orders to be fulfilled;
- (ii) schedule the lots to minimize some objective function measuring the deviation from due-dates.

Problem (i) will be here referred to as the ITEM-TO-LOT ASSIGNMENT PROBLEM (ILAP). To illustrate the ILAP, consider the 12 patterns of Fig. 1, derived from a real-world application. They all have run length 1, except pattern 7, 10, 12. Two lots are shown: 1 (red), 2 (green). Cut sequence clearly affects lot completion time, but this also depends on what patterns feed what lots: in the example, three type 4 items are cut according to pattern 10 and sent to lot 2, whereas one item of the same type comes from pattern 11 and feeds lot 1; the same pattern sequence would terminate the lots in different instants, should type 4 item for lot 1 be produced by pattern 10.

In principle, the choice of the solution approach depends on goal priorities or shop-floor practice, as well as on problem size and computational resources available. The computational burden of the integrated approach, as well as the fact that the CSP solution algorithms used in the industrial practice are black-boxes not accessible by users, make pattern sequencing an obliged choice in general industrial practice. Nevertheless, compared to other problems in the cutting stock literature, the PSP and the problems derived from its decomposition have not yet received considerable attention to the best of our knowledge.

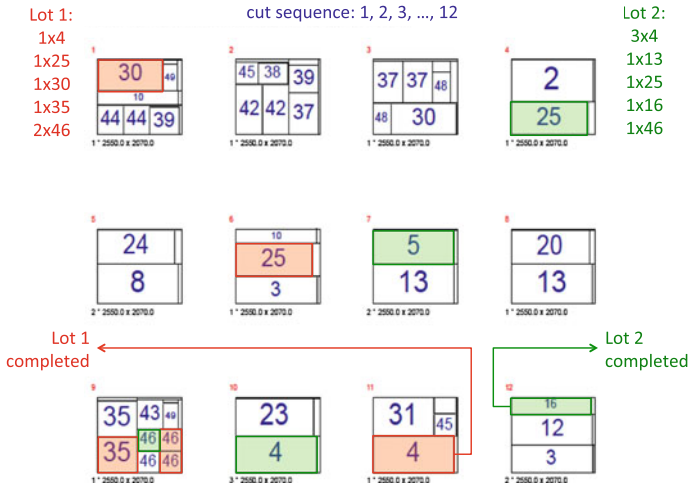


Fig. 1 Item-to-lot assignment in a pattern set

In this paper, we focus on the PSP and survey a few optimization models for finding the best schedule of a given set of cuts. In Sect. 2 we present a model for the ILAP with the objective of controlling how orders are spread through patterns: its preliminary solution provides an input for a subsequent PSP where each item has a non-ambiguous due-date, namely the one of the lot the item belongs to. The general PSP that integrates item-to-lot assignment and pattern sequencing is addressed in Sect. 3.

## 2 Item-to-lot Assignment and Order Spread

This problem arises when scheduling the  $n$  cut operations obtained as a solution of a cutting stock problem. Each operation produces different parts, and those parts are generally required in heterogeneous lots by downstream departments.

Specifically, suppose to produce  $r$  different part types and consider the problem of sequencing patterns to optimize a due-date related objective. A decision is to be made between pattern construction and scheduling: in fact, part type  $k$  is produced by various patterns and required by various lots, so one has to determine which pattern feeds which lot. For example, part type  $k$ , required in amounts  $b_{k\alpha}, b_{k\beta}$  by lots  $\alpha$  and  $\beta$ , is produced in quantities  $a_k^p, a_k^q$  by patterns  $p, q$ . Since supply and demand are balanced, we have  $a_k^p + a_k^q = b_{k\alpha} + b_{k\beta}$ , but we still have to decide what amount of supply from patterns  $p$  and  $q$  will feed lot  $\alpha$  and, therefore, lot  $\beta$ .

This decision affects lots' minimum processing times (what we here call *lot spread*), because a lot is finished as soon as all the cuts of the patterns that feed it are

completed: therefore, the decision indirectly affects the quality of pattern scheduling. What is a good spread then? At a first approach, it appears reasonable that the less urgent lots are those getting a larger spread, and vice-versa.

To formulate the ILAP, let

$I$  be the set of the small objects to be produced ( $|I| = m$ );  $i \in I$  denotes an individual item, that is, identical items of the same type (shape, size, material etc.) receive different indexes.

$R$  be the set of part types ( $|R| = r$ ):  $I$  is partitioned into subsets  $I_1, \dots, I_r$ , with  $I_k$  containing all the items of type  $k$  required.

$L$  be the set of all the lots to be produced to fulfill an order ( $|L| = l$ ); a lot  $\ell \in L$  is a heterogeneous collection of items of  $I$ , and is represented as an integer  $r$ -vector  $\mathbf{b}_\ell$ , where  $b_{k\ell}$  is the number of type  $k$  parts lot  $\ell$  requires;  $\ell \in L$  may be associated with a due-date  $d_\ell$ .

$P$  be the set of patterns ( $|P| = n$ ): each  $p \in P$  is just an operation that produces  $a_k^p$  items of type  $k \in R$ , therefore it is represented by an integer  $r$ -vector  $\mathbf{a}^p$ .

$T$  be the set of time slots ( $|T| = n$ ).

Let  $x_{k\ell}^p$  be the integer variable measuring the number of type  $k$  parts that pattern  $p$  sends to feed lot  $\ell$ . The pattern capacity (number  $a_k^p$  of type  $k$  parts produced by pattern  $p$ ) must be respected:

$$\sum_{\ell \in L} x_{k\ell}^p = a_k^p \quad \forall p \in P, k \in R \quad (1)$$

The demand  $b_{k\ell}$  of parts of type  $i$  from lot  $\ell$  must be fulfilled:

$$\sum_{p \in P} x_{k\ell}^p = b_{k\ell} \quad \forall k \in R, \ell \in L \quad (2)$$

with  $\sum_p a_k^p = \sum_\ell b_{k\ell}$ .

As noticed, in an ideal lot feeding, urgent lots are as less as possible spread over patterns. One possibility is to minimize the largest time required to operate the patterns that feed each single lot, weighted according to lot urgency. Using 0–1 variables  $y_\ell^p$  to indicate that pattern  $p$  feeds lot  $\ell$ , hence we must first enforce

$$x_{k\ell}^p \leq \min\{a_k^p, b_{k\ell}\} y_\ell^p \quad \forall k \in R, \ell \in L, p \in P \quad (3)$$

then bind the total minimum feeding time from above

$$w_\ell \sum_{p \in P} t^p y_\ell^p \leq s \quad \forall \ell \in L \quad (4)$$

and finally minimize the weighted spread  $s$ . In the last formula,  $w_\ell$  is a weight that increases with lot due-date  $d_\ell$ , and  $t^p$  denotes the time spent to process pattern  $p$ .

### 3 Pattern Sequencing

To formulate a pattern sequencing problem that integrates item-to-lot assignment, one can start from Gramani et al. coupled cutting stock and lot sizing model, keeping in mind that, unlike [6], the pattern set is in our case given.

In our case we do not consider aggregate demand, production and stock of finished products. This position reflects make-to-order productions as in the furniture industry, where each final product generally corresponds to a single order with an individual due-date. Order setup and holding cost are however taken into account, as in [6].

Our formulation assigns items to lots, and patterns and lots to time periods. To express this and the decisions regarding lot feeding and scheduling, we use the following main 0–1 decision variables:

- $x^{pt} = 1$  iff pattern  $p \in P$  is scheduled at time  $t \in T$
- $y_{i\ell} = 1$  iff item  $i \in I$  is sent to lot  $\ell \in L$
- $z_{\ell}^t = 1$  iff lot  $\ell \in L$  is fed (by some pattern) at time  $t \in T$ .

These variables are subject to the constraints described in the following.

#### 3.1 Main Constraints

The  $x$ -variables define pattern permutations and are thus subject to matching equations:

$$\begin{aligned} \sum_{t \in T} x^{pt} &= 1 & \forall p \in P \\ \sum_{p \in P} x^{pt} &= 1 & \forall t \in T \end{aligned} \tag{5}$$

i.e., every pattern must be scheduled at some time, and no two patterns can be scheduled at the same time. On the other hand, the  $y$ -variables are bound to an equivalent of lot demand condition (2) and to item assignment conditions

$$\begin{aligned} \sum_{i \in I_k} y_{i\ell} &= b_{k\ell} & \forall \ell \in L, k \in R \\ \sum_{\ell \in L} y_{i\ell} &= 1 & \forall i \in I \end{aligned} \tag{6}$$

which ensure that every lot  $\ell$  gets the required amount of parts of type  $k$  produced and that every item is assigned to exactly one lot.

Then, both  $x$ - and  $y$ -variables are involved in an implication on lot timing:

$$x^{p(i),t} + y_{i\ell} - z_{\ell}^t \leq 1 \quad \forall i \in I, \ell \in L, t \in T \tag{7}$$

where  $p(i)$  is the pattern that produces item  $i$ : meaning that if item  $i$  is assigned to lot  $\ell$  and is produced at time  $t$ , then lot  $\ell$  is still fed at time  $t$ .

### 3.2 Objectives

To write the objective function, we need auxiliary variables. One is the completion time  $C_\ell$  of lot  $\ell$ , which cannot be less than the last time instant at which  $\ell$  is fed:

$$C_\ell \geq z_\ell^t \quad \forall \ell \in L, t \in T \quad (8)$$

(in constraint (8) we assume that each pattern is cut in unit time). Two more variables  $u_\ell^t, v_\ell^t$  define the time span in which  $\ell$  is fed with parts, and are subject to:

$$u_\ell^{t-1} \geq u_\ell^t \geq z_\ell^t \quad v_\ell^{t+1} \geq v_\ell^t \geq z_\ell^t \quad (9)$$

where  $u_\ell^t = 1$  ( $v_\ell^t = 0$ ) indicates that  $\ell$  is not completed (not started) yet at time  $t$ . The  $u$ - and  $v$ -variables can be mutually constrained as explained in Sect. 3.3.

Various objectives can be written via the variables and constraints introduced:

#### Total weighted completion time

$$C_{tot} = \sum_{\ell \in L} w_\ell C_\ell \quad (10)$$

#### Total weighted tardiness

$$T_{tot} = \sum_{\ell \in L} w_\ell T_\ell \quad (11)$$

with  $T_\ell \geq C_\ell - d_\ell$  for all  $\ell \in L$ .

#### Weighted number of tardy jobs

$$J_{tot} = \sum_{\ell \in L} w_\ell \eta_\ell \quad (12)$$

with  $\eta_\ell \geq z_\ell^t$  for  $t > d_\ell$  and all  $\ell \in L$ .

#### Maximum lateness

$$L_{max} \geq C_\ell - d_\ell \quad (13)$$

for all  $\ell \in L$ .

#### Maximum number of unfinished lots (open stacks)

$$O_{max} \geq \sum_{\ell \in L} (u_\ell^t + v_\ell^t - 1) \quad (14)$$

for all  $t \in T$ .

**Total inventory (holding) cost**

$$H_{tot} = \sum_{\ell \in L} \sum_{t \in T} h_{\ell}^t (u_{\ell}^t + v_{\ell}^t - 1) \tag{15}$$

where  $h_{\ell}^t$  is the cost of holding lot  $\ell \in L$  during period  $t \in T$ .

**3.3 Valid Inequalities**

Let  $R_{\ell}$  denote the set of part types required by lot  $\ell$ , and  $\tau$  (a lower bound to) the value of an optimal solution of the following cutting stock problem:

$$\begin{aligned} \tau \leq \min \sum_{p \in P} x^p & \tag{16} \\ \sum_{p \in P} a_k^p x^p \geq b_{k\ell} & \quad k \in R_{\ell} \\ x^p \in \{0, 1\} & \quad p \in P \end{aligned}$$

In the above formulation,  $a_k^p$  is the number of parts of type  $k$  produced with pattern  $p$  at full run length. Then  $\ell$  cannot be completed before  $\tau$  time periods, which means

$$v_{\ell}^t - v_{\ell}^{t-1} - u_{\ell}^{t+\tau} \leq 0 \quad \forall t \leq |T| - \tau \tag{17}$$

In fact, if  $\ell$  begins at time  $t$ , then it is active at time  $t$  and not  $t - 1$ : in this case the first two terms of inequality (17) sum up to 1, thus implying  $\ell$  active at time  $t + \tau$ .

**3.4 Formulation Size**

In typical applications there are  $n = 30$  to 100 patterns that produce  $m = 100$  to 1,000 items of  $r = 50$  to 100 different types to feed  $l = 10$  to 20 lots. The table of Fig. 2 reports an indication of the number of main variables and constraints with  $n$  ranging between 30 and 100. We suppose that each pattern produces on average 10 items, for a total amount that ranges from 300 to 1,000. The number  $l$  of lots varies between 6 and 20: each lot is assumed to contain around 50 items of 5 distinct types. Figure 2 reports the relevant numbers of variables and constraints.

n	items per pattern	m	items per lot	l	types per lot	variables				constraints				
						x	y	z	total	(5)	(6)	(7)	(8)	total
30	10	300	50	6	5	9.000	1.800	180	10.980	60	330	54.000	180	54.570
40	10	400	50	8	5	16.000	3.200	320	19.520	80	440	128.000	320	128.840
50	10	500	50	10	5	25.000	5.000	500	30.500	100	550	250.000	500	251.150
60	10	600	50	12	5	36.000	7.200	720	43.920	120	660	432.000	720	433.500
70	10	700	50	14	5	49.000	9.800	980	59.780	140	770	686.000	980	687.890
80	10	800	50	16	5	64.000	12.800	1.280	78.080	160	880	1.024.000	1.280	1.026.320
90	10	900	50	18	5	81.000	16.200	1.620	98.820	180	990	1.458.000	1.620	1.460.790
100	10	1.000	50	20	5	100.000	20.000	2.000	122.000	200	1.100	2.000.000	2.000	2.003.300

Fig. 2 An indication of the growth of problem size with problem parameters

### 4 Conclusions

In this paper we have introduced two optimization problems to schedule the cuts resulting from the solution of a cutting-stock problem (CSP), namely the Pattern Sequencing Problem (PSP) and a derived subproblem, called Item-to-Lot Assignment Problem (ILAP). Both the optimization problems are of great interest in the industrial practice. For each of them, we have discussed integer programming formulations, to be used as starting points to devise IP-based solution algorithms.

**Acknowledgements** Work supported by the Italian Ministry of Education, National Research Program (PRIN) 2015, contract no. 20153TXRX9.

### References

1. Arbib, C., Marinelli, F.: On cutting stock with due dates. *Omega Int. J. Manag. Sci.* **46**, 11–20 (2014)
2. Arbib, C., Marinelli, F.: Maximum lateness minimization in one-dimensional bin packing. *Omega Int. J. Manag. Sci.* **68**, 76–84 (2017)
3. Arbib, C., Felici, G., Servilio, M.: Sorting common operations to minimize the number of tardy jobs. *Networks* **64**(4), 306–320 (2014)
4. Arbib, C., Marinelli, F., Ventura, P.: One-dimensional cutting stock with a limited number of open stacks: bounds and solutions from a new integer linear programming model. *Int. Trans. Oper. Res.* **23**(1–2), 47–63 (2016)
5. Belov, G., Scheithauer, G.: Setup and open-stacks minimization in one-dimensional stock cutting. *INFORMS J. Comput.* **19**(1), 27–35 (2007)
6. Gramani, M.C.N., Franca, P.M., Arenales, M.N.: A Lagrangian relaxation approach to a coupled lot-sizing and cutting stock problem. *Int. J. Prod. Econ.* **119**, 219–227 (2009)



7. Russo, M., Sforza, A., Sterle, C.: An exact dynamic programming algorithm for large-scale unconstrained two-dimensional guillotine cutting problems. *Comput. Oper. Res.* **50**, 97–114 (2014)
8. Smith, B.M., Gent, I.P.: Constraint modelling challenge 2005. In: *Fifth Workshop on Modelling and Solving Problems with Constraints*, IJCAI 2005, Edinburgh, Scotland, 31 July 2005

# Bin Packing Problems with Variable Pattern Processing Times: A Proof-of-concept

Fabrizio Marinelli and Andrea Pizzuti

**Abstract** In several real-world applications the time required to accomplish a job generally depends on the number of tasks that compose it. Although the same also holds for packing (or cutting) problems when the processing time of a bin depends by the number of its items, the approaches proposed in the literature usually do not consider variable bin processing times and therefore become inaccurate when time costs are worth more than raw material costs. In this paper we discuss this issue by considering a variant of the one-dimensional bin packing problem in which items are due by given dates and a convex combination of number of used bins and maximum lateness has to be minimized. An integer linear program that takes into account variable pattern processing times is proposed and used as proof-of-concept.

**Keywords** One-dimensional bin packing · Scheduling  
Mixed integer programming

## 1 Introduction

The general bin-packing problem (BP) consists of assigning a set of items to identical bins in order to minimize the number of filled bins. A solution of the BP is feasible if each item is packed without overlapping with any other and it is completely contained inside a bin. A large literature focuses on the study of the BP and its extensions, see [11].

---

F. Marinelli (✉) · A. Pizzuti (✉)  
Dipartimento di Ingegneria dell'Informazione, Università Politecnica  
delle Marche, Via Breccie Bianche, 60131 Ancona, Italy  
e-mail: fabrizio.marinelli@univpm.it

A. Pizzuti  
e-mail: a.pizzuti@pm.univpm.it

© Springer International Publishing AG 2017  
A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_46

Due to its mathematical structure, the BP arises in many real applications among which cutting optimization in manufacturing, loading operations in logistics, data packets allocation in telecommunication and advertisement scheduling in publicity [6]. It is also well-known that the BP can be regarded as a scheduling problem in which each item corresponds to a job that consumes the availability of some resources represented by the bins [7].

If a time dimension is also considered and each item is provided with a specific due-date, then a solution of the BP can be generally understood as a sequence of bins each one processed at a prescribed time instant. Then, a multi-criteria problem arises if additional scheduling objectives, i.e. functions of the completion times, are taken into account, such as the sum of (weighted) completion times, the maximum lateness, the (weighted) tardiness or the number of tardy jobs.

A growing number of papers in the last years addresses the effects of scheduling issues in the BP and cutting stock problems. Among them, both exact [2, 3] and heuristic [4, 10] approaches are proposed for different scheduling objectives and BP variants. However, two common assumptions are made about the packing (or cutting) process: the packing time is constant and independent from the structure of the pattern, and the items of a bin are all released only once the packing of the whole bin is completed. To the best of our knowledge, only in [5] the authors explicitly treat the reduction of the number of cuts to minimize the cutting costs of a three-staged cutting problem; nevertheless, time is worth more than material waste in many real cutting and packing industrial processes, and in all such cases potential variable processing time of bins should be specifically addressed. Meaningful examples can be found in paper or steel manufacturing, where the minimization of the used raw material can be formulated as a one-dimensional BP. Generally, it is supposed that machines are equipped with a set of automatic slitting knives and each pattern is performed with a single parallel cut in constant cutting time. Still, due to the particular application, it is not unlikely to have machines equipped with a single manual slitting knife. In these cases, each bar (or roll) is sequentially cut into slices (accordingly to the prescribed pattern): each item is therefore available as soon as it is processed, and the total time required to process the bar strictly depends by the number of cuts performed, that is by the number of slices that compose the pattern.

This paper is a first attempt to explicitly take into account patterns with variable processing times in the context of the BP with scheduling issues. In particular, we address the one-dimensional bin packing problem (1BP-VPT) where each item is provided by a due-date and is processed in constant time, and propose a mixed integer linear programming formulation (MILP) that minimizes a convex combination of the number of used bins and the maximum lateness.

In Sect. 2 the problem 1BP-VPT is formalized, and the MILP formulation is described. In Sect. 3 we report some computational results and discuss the outcomes implied by the introduction of variable processing times; finally, in Sect. 4 some conclusions are depicted.

## 2 Problem Definition and Model Formulation

In the classic one-dimensional BP problem, the set  $I = \{1, \dots, n\}$  of items is described by positive integer lengths  $l_1, \dots, l_n$ , and the bins have identical integer size  $W$  such that  $l_i \leq W, i = 1, \dots, n$ . The purpose is to find a feasible solution in which all the items are packed within the minimum number  $z^*$  of bins.

In 1BP-VPT each item  $i \in I$  is also due by a given date  $d_i$  and requires a constant time  $t$  to be packed. Moreover, each bin requires a constant time  $s$  to be set-up for the packing, as assumed in the cutting stock problem with the number of patterns minimization [1, 12]. A solution of 1BP-VPT consists of a set of packing patterns and the sequence in which the patterns are applied to bins. Let  $q_i$  be the position of the bin (in the sequence of patterns) that contains the item  $i$ , and  $p_i$  be the absolute position of the item  $i$  in the sequence of the packed items, i.e.,  $p_i = k$  if the total number of items packed before  $i$ , from the first bin on, is  $k - 1$ . Then, the completion time  $C_i$  of the item  $i$  is given by  $sq_i + tp_i$  and its lateness is  $L_i = C_i - d_i$ . The objective function of 1BP-VPT is  $F = \alpha_1 z + \alpha_2 L_{max}$ , with  $\alpha_1, \alpha_2 \geq 0$  and  $\alpha_1 + \alpha_2 = 1$ , that is a convex combination of the number  $z$  of used bins and the maximum lateness  $L_{max} = \max_{1 \leq i \leq n} L_i$ . Parameters  $\alpha_1$  and  $\alpha_2$  describe the relative ratio between material and delay costs, and their value is strictly application-dependent. The definition of  $C_i$  describes the fact that the item  $i$  is available as soon as it is processed, instead of at the completion time of the whole bin in which it is packed. Actually, a solution of 1BP-VPT describes a sequence of items instead of a sequence of patterns.

Aside from its practical usefulness, the choice of  $L_{max}$  as reference scheduling term has been suggested by the small *elasticity* that it exhibits with respect to the items processing time  $t$ . Indeed,  $L_{max}$  only depends by the item with the largest delay, while the value of other scheduling objective functions generally takes into account the delay of each item. A small example clarifies such behaviour: let  $I$  be the set of four items with lengths  $\{l_1, l_2, l_3, l_4\} = \{6, 4, 3, 5\}$  and due-dates  $\{d_1, d_2, d_3, d_4\} = \{1, 1, 1, 2\}$ . Let moreover  $W = 10, s = 1$  and let us consider

- the total tardiness  $F_T = \sum T_i$ , where  $T_i = \max\{0, L_i\}$ , and
- the number of tardy jobs  $F_U = \sum U_i$ , where  $U_i = \begin{cases} 1 & \text{if } C_i > d_i \\ 0 & \text{otherwise} \end{cases}$ .

Any optimal solution requires two bins and, for  $t = 0, L_{max} = F_T = F_U = 1$ , since one of the items 2 or 3 must be packed in the second bin. On the other hand, for  $t = 0.2$ , the value of  $L_{max}$  just increases to 1.6, whereas  $F_T$  is tripled and  $F_U$  quadrupled, and moreover the gap between  $L_{max}$  and  $F_T$  grows for increasing values of  $t$ .

Let  $J = \{1, \dots, m\}$  denote the set of available bins, with  $m = n$ . For each  $i \in I$  and  $j \in J$ , let us define the following decision variables:

- $x_{ij} \in \{0, 1\}$ :  $x_{ij} = 1$  if and only if the item  $i$  is assigned to the  $j$ -th bin;
- $y_j \in \{0, 1\}$ :  $y_j = 1$  if and only if the  $j$ -th bin is used;
- $q_i \in \mathbb{N} \setminus \{0\}$ :  $q_i = j$  if and only if the  $j$ -th bin contains the item  $i$ ;

- $p_i \in \mathbb{N} \setminus \{0\}$ :  $p_i = k$  if and only if the item  $i$  is processed as  $k$ -th item of the sequence of packed items;
- $\pi_{ih} \in \{0, 1\}$ :  $\pi_{ih} = 1$  if and only if the item  $h$  is processed before the item  $i$ .

We propose the following MILP formulation ( $M_{VPT}$ ):

$$\min F = \alpha_1 z + \alpha_2 L_{max} \tag{1}$$

$$\sum_{j=1}^m x_{ij} = 1 \quad \forall i \in I \tag{2}$$

$$\sum_{i=1}^n l_i x_{ij} \leq W y_j \quad \forall j \in J \tag{3}$$

$$z = \sum_{j=1}^m y_j \tag{4}$$

$$q_i = \sum_{j=1}^m j x_{ij} \quad \forall i \in I \tag{5}$$

$$C_i = s q_i + t p_i \quad \forall i \in I \tag{6}$$

$$C_i - d_i \leq L_{max} \quad \forall i \in I \tag{7}$$

$$q_i - q_h \leq (m - 1) \pi_{ih} \quad \forall i, h \in I : i \neq h \tag{8}$$

$$q_h - q_i \leq (m - 1)(1 - \pi_{ih}) \quad \forall i, h \in I : i \neq h \tag{9}$$

$$p_i - p_h + 1 \leq n \pi_{ih} \quad \forall i, h \in I : i \neq h \tag{10}$$

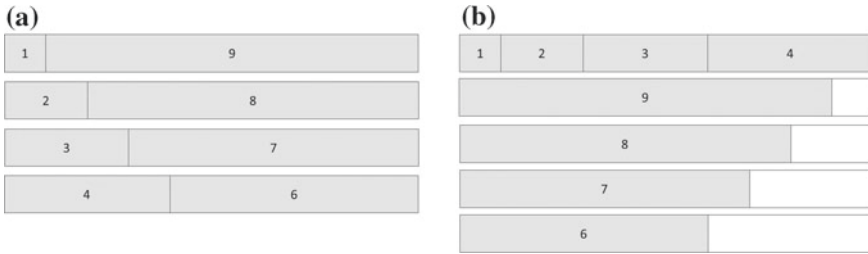
$$p_h - p_i + 1 \leq n(1 - \pi_{ih}) \quad \forall i, h \in I : i \neq h \tag{11}$$

$$x_{ij}, y_j \in \{0, 1\} \quad \forall i \in I, j \in J \tag{12}$$

$$q_i, p_i \in \mathbb{N} \setminus \{0\} \quad \forall i \in I \tag{13}$$

$$\pi_{ih} \in \{0, 1\} \quad \forall i, h \in I : i \neq h \tag{14}$$

Constraints (2)–(4) belong to the well-known assignment formulation [8] for one-dimensional BP: (2) states that every item must be packed; (3) ensures that items do not overlap and are completely contained within the bins; (4) defines the BP objective function. Equation (5) expresses the position of the bin that contains item  $i$ , Eq. (6) defines the completion time of item  $i$ , and constraint (7) bounds from below the maximum lateness. Disjunctive inequalities (8)–(11) model the relationship between the sequence of the bins and the processing order of the items: if the bin that contains  $i$  strictly precedes the bin that contains  $h$ , then  $i$  must have been processed before  $h$ ; on the opposite, if  $i$  is processed before  $h$ , then  $i$  is packed in the same bin of  $h$  or in a previous one; finally, if  $i$  and  $h$  are packed in the same bin, then a strict ordering of the respective positions is guaranteed.



**Fig. 1** a Optimal for  $t = 0$  and sub-optimal for  $t = 1$ ; b optimal for  $t = 1$  and sub-optimal for  $t = 0$

Optimal solutions of 1BP-VPT are inherently different from those obtained by considering constant processing times of bins. Indeed, the following example highlights that a solution which is optimal for  $t = 0$ , can be non-optimal for  $t > 0$ , i.e., when bin processing time is not constant, and vice-versa. Let  $I = \{1, 2, \dots, 8\}$  be the items set with lengths  $\{l_1, l_2, l_3, l_4, l_5, l_6, l_7, l_8\} = \{1, 2, 3, 4, 6, 7, 8, 9\}$  and due-dates  $\{d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8\} = \{1, 2, 3, 4, 8, 7, 6, 5\}$ . Let moreover assume  $W = 10$ ,  $s = 1$  and  $\alpha_1 = \alpha_2 = 0.5$ , with  $t = 0$  in first place. Figure 1 shows two solutions in which the sequence of bins (items) proceeds from top (left) and labels refer to the item lengths. It is easy to see that solution a) with  $z = 4$ ,  $L_{max} = 0$  and  $F = 2$  is optimal, whereas solution b) with  $z = 5$ ,  $L_{max} = 0$  and  $F = 2.5$  is not. On the contrary, for  $t = 1$ , solution (a) is no longer optimal since it has  $z = 4$ ,  $L_{max} = 7$  defined by item 4 and  $F = 5.5$ , whereas solution (b) with  $z = 5$ ,  $L_{max} = 5$  defined by item 5 and  $F = 5$  becomes optimal.

In this case the optimality changes to the looseness of the due-dates: for  $t = 0$ , the number of bins is critical, while the lateness are not binding; for  $t = 1$ , the relevance of the lateness increases over the number of bins weight and the use of an additional bin is required to redefine the items ordering and reduce the lateness value. Generally speaking, the introduction of item processing times modifies the balance among the objective function terms in such a way that optimal solutions for  $t = 0$  significantly differ from the truly optimal value of  $F$ . Therefore, formulations that neglect the item processing times may lead to sub-optimal solutions with meaningful optimality gaps.

### 3 Preliminary Experiments

Some preliminary tests were made on  $M_{VPT}$  for 1BP-VPT and comparison has been made with the time-indexing MILP ( $M_K$ ) described in [3], a basic formulation to solve the same packing and scheduling problem but with constant processing time of patterns. Formulations were implemented with AMPL (version 20150214, MS

VC++ 6.0, 32-bit) and solved by Cplex 12.5.0.0 with default setting. Tests were carried out on an Intel® Core™ i3 M350 2.27 GHz with 4 GB RAM.

Experiments were made on fifty instances with  $n = 10$ ,  $W = 1000$ ,  $l_i$  randomly chosen in  $[1, W]$  for  $1 \leq i \leq n$ , and integer due-dates randomly chosen in  $[1, [(s + tn)LB_C]]$ , where  $LB_C$  is the continuous lower bound for the classic one-dimensional BP [9]. The  $L_{max}$  value of the solutions achieved by  $M_K$  has been adjusted in order to take into account the processing time  $t$  in the computation of completion times. Finally, we used  $s = 1$  and  $t = 0.2$ , and run two scenarios by setting  $\alpha_1 = 0.5$  and  $\alpha_1 = 0.1$ , respectively. Tables 1 and 2 report the results of the two scenarios aggregated in five classes, each one consisting of 10 instances. For each class, the mean values of  $z$  and  $L_{max}$  obtained by  $M_{VPT}$  and  $M_K$  are listed. Average running times are also reported.

Results show that the  $L_{max}$  term of the optimal solutions computed by  $M_K$  has a significant relative error with respect to the optimal values of 1BP-VPT (33.3% for  $\alpha_1 = 0.5$  and 35.7% for  $\alpha_1 = 0.1$ , on average). We also observed that such relative error increases as far as the due-dates get loose, i.e., the solutions for  $t = 0$  are more oriented to the minimization of the number  $z$  of bins. As showed by Table 2, the effect of neglecting  $t$  is generally more evident when  $\alpha_2$  increases. Indeed, when

**Table 1**  $M_{VPT}$  and  $M_K$  results for  $\alpha_1 = \alpha_2 = 0.5$

Class	$M_{VPT}$			$M_K$		
	$z$	$L_{max}$	CPU time (s)	$z$	$L_{max}$	CPU time (s)
I	4.6	1.08	0.663	4.6	1.72	0.020
II	4.5	1.66	0.803	4.5	2.20	0.214
III	4.7	1.64	0.780	4.7	2.12	0.045
IV	4.5	1.78	0.944	4.5	2.10	0.115
V	5.1	1.64	0.477	5.1	2.26	0.120
Average	4.7	1.56	0.734	4.7	2.08	0.103

**Table 2**  $M_{VPT}$  and  $M_K$  results for  $\alpha_1 = 0.1, \alpha_2 = 0.9$

Class	$M_{VPT}$			$M_K$		
	$z$	$L_{max}$	CPU time (s)	$z$	$L_{max}$	CPU time (s)
I	4.8	1.04	0.629	4.6	1.86	0.148
II	4.5	1.66	0.785	4.5	2.10	0.229
III	4.7	1.64	0.839	4.7	2.34	0.268
IV	4.7	1.72	0.941	4.5	2.00	0.165
V	5.2	1.62	0.471	5.1	2.08	0.254
Average	4.8	1.54	0.733	4.7	2.08	0.213

$L_{max}$  gains weight, solutions with additional number of bins become attractive and optimal solutions become more related to the delay reduction.

About the running times,  $M_{VPT}$  computed optimal solutions in 0.734 s. (0.733 s.) on average, whereas  $M_K$  takes just 0.103 s. (0.213 s.) on average. Beside the larger size of  $M_{VPT}$ , the main reason of such worsening clearly lies in the presence of symmetries on variables  $\pi_{ij}$  and disjunctive constraints (8)–(11).

## 4 Conclusions

In several real-world packing (or cutting) applications, items are sequentially processed and the total time needed to realize a pattern depends by the number of items that form it. When time is worth more than some other resource costs, item processing times may considerably affect the solution costs, but available formulations for BP usually do not consider variable pattern processing times and therefore often become inaccurate for computing optimal solutions. In this paper we discussed such issue and proposed a basic MILP formulation to solve 1BP-VPT, a one-dimensional bin packing problem where the item processing times are explicitly considered and the minimization of a convex combination of the number of filled bins and the maximum lateness value is required. Some preliminary experiments show that the difference between the optimal solution values of 1BP-VPT and those obtained by considering constant pattern processing times is enough to highlight the relevance of the issue being discussed.

**Acknowledgements** Work supported by the Italian Ministry of Education, National Research Program (PRIN) 2015, contract n. 20153TXRX9.

## References

1. Aloisio, A., Arbib, C., Marinelli, F.: On LP relaxations for the pattern minimization problem. *Networks* **57**(3), 247–253 (2011)
2. Arbib, C., Marinelli, F.: On cutting stock with due dates. *Omega Int. J. Manag. Sci.* **46**, 11–20 (2014)
3. Arbib, C., Marinelli, F.: Maximum lateness minimization in one-dimensional bin packing. *Omega Int. J. Manag. Sci.* **68**, 76–84 (2017)
4. Bennel, J.A., Lee, L.S., Potts, C.N.: A genetic algorithm for two-dimensional bin packing with due dates. *Int. J. Prod. Econ.* **145**, 547–560 (2013)
5. Cui, Y., Huang, B.: Reducing the number of cuts in generating three-staged cutting patterns. *Eur. J. Oper. Res.* **218**, 358–365 (2012)
6. Delorme, M., Iori, M., Martello, S.: Bin packing and cutting stock problems: mathematical models and exact algorithms. *Eur. J. Oper. Res.* **255**, 1–20 (2016)
7. Hartmann, S.: Packing problems and project scheduling models: an integrating perspective. *J. Oper. Res. Soc.* **51**, 1083–1092 (2000)
8. Kantorovich, L.V.: Mathematical methods of organizing and planning production. *Manag. Sci.* **6**, 366–422 (1960)



9. Martello, S., Toth, P.: Lower bounds and reduction procedures for the bin packing problem. *Discret. Appl. Math.* **28**, 59–70 (1990)
10. Reinertsen, H., Vossen, T.W.M.: The one-dimensional cutting stock problem with due-dates. *Eur. J. Oper. Res.* **201**, 701–711 (2010)
11. Wäscher, G., Haußner, H., Schumann, H.: An improved typology of cutting and packing problems. *Eur. J. Oper. Res.* **183**, 1109–1130 (2007)
12. Yanasse, H.H., Limeira, M.S.: A hybrid heuristic to reduce the number of different patterns in cutting stock problems. *Comput. Oper. Res.* **33**, 2744–2756 (2006)

# Upper Bounds Categorization for Constrained Two-Dimensional Guillotine Cutting

Mauro Russo, Antonio Sforza and Claudio Sterle

**Abstract** In the two-dimensional cutting problem, a large rectangular sheet has to be dissected into smaller rectangular desired pieces. If limits exist on the number of extracted pieces, the problem is classified as constrained, with a wide range of applications. Most literature solving methods are based on ad hoc tree search strategies, with top-down or bottom-up approach. In both cases, lower and upper bounds are exploited, leading to branch and bound algorithms. We present a review of the upper bounds and identify a set of features for their categorization.

**Keywords** Guillotine cutting · Upper bound categorization

## 1 Problem Formulation and Solution Methods

In the class of cutting and packing (C&P) problems, a strong interest has been shown in literature for the two-dimensional problems. Main applications are in the production of materials, for which the constraint of guillotine cuts is often needed.

We consider the problem where a large rectangular object (sheet or plate) has to be dissected to extract rectangular piece types, with guillotine cuts. According to [34], this problem is classified as guillotine variant of 2-dimensional rectangular SLOPP (Single Large Object Placement Problem), but it is often referred to as 2-Dimensional Cutting, or 2DC/TDC (e.g. in [12]). This problem is NP-hard [22].

---

M. Russo (✉)

Intecs Solutions, via Ferrante Imparato 198, 80146 Naples, Italy  
e-mail: mauro.russo@intecs.it

A. Sforza · C. Sterle

Department of Electrical Engineering and Information Technology,  
University “Federico II” of Naples, via Claudio 21, 80125 Naples, Italy  
e-mail: sforza@unina.it

C. Sterle

e-mail: claudio.sterle@unina.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_47

In case of limited piece demands, the problem is *constrained* (C2DC), otherwise *unconstrained* (U2DC) if all demands are unlimited. According to the relation between profit and area of the pieces, the problem is *weighted* or *unweighted*. Most authors use a fixed orientation, i.e. with no chance to rotate pieces.

We treat the C2DC, both weighted and unweighted, with fixed orientation. In the following we formalize the problem and describe the existing methods to solve it. In Sect. 2 we present four classification criteria for related upper bounds. In Sect. 3 we match the upper bounds with a proper list of literature papers.

A rectangular sheet  $(L, W)$  and  $n$  piece types  $(t_1, \dots, t_n)$  are given. Each type  $t_i$  has dimensions  $(l_i, w_i)$ , demand  $d_i$  and profit  $\pi_i$ . We refer to the ratio  $\pi_i / (l_i \cdot w_i)$  as *profitability*. The problem is unweighted if  $\pi_i = l_i \cdot w_i$  for all  $i$ . The pieces cannot be rotated to  $(w_i, l_i)$ . A vector  $B = (b_1, \dots, b_n)$  represents a set of pieces with frequencies  $b_i$  and profit  $\pi(B) = b_1 \cdot \pi_1 + \dots + b_n \cdot \pi_n$ . It is a *cutting pattern* if its pieces can be extracted using guillotine cuts. It is *feasible* if  $b_i \leq d_i$  for all  $i$ . All parameters are positive integers. The goal is to maximize  $\pi(B)$  over all feasible patterns.

### 1.1 Dynamic Programming Methods

The *knapsack function* [15]  $f_U$  gives the best profit in the unconstrained case for any rectangle  $(l, w)$ . It is computed in recursion (1)–(2.a, 2.b, 2.c) as best among  $f_{U,0}$  (one piece term),  $f_{U,v}$  or  $f_{U,h}$  (vertical/horizontal cut, i.e. horizontal/vertical merge).

$$f_U(l, w) = \max\{f_{U,0}(l, w), f_{U,v}(l, w), f_{U,h}(l, w)\}, 0 \leq l \leq L; 0 \leq w \leq W \quad (1)$$

$$f_{U,0}(l, w) = \max\{0; \pi_i: l_i \leq l, w_i \leq w\} \quad (2.a)$$

$$f_{U,v}(l, w) = \max\{0; f_U(x_1, w) + f_U(x_2, w): 0 < x_1 \leq x_2, x_1 + x_2 = l\} \quad (2.b)$$

$$f_{U,h}(l, w) = \max\{0; f_U(l, y_1) + f_U(l, y_2): 0 < y_1 \leq y_2, y_1 + y_2 = w\} \quad (2.c)$$

Two implementations exist. We refer to them as bottom-up and top-down respectively, as used in literature for the tree search solution methods. In the first case, an outer loop scrolls all couples  $(x_2, y_2)$  in increasing order (one line per time). Two independent inner loops, on  $x_1$  and  $y_1$  respectively, generate horizontal or vertical merging, with dimensions  $(x_1 + x_2, y_2)$  or  $(x_2, y_1 + y_2)$ . This approach has been followed in [15] and in recent improvements [28, 29]. In the second case, the couples  $(l, w)$  scrolled at the outer loop represent the dimensions of a dissected rectangle, whereas  $x_1$  and  $y_1$  in the inner loops correspond to cut coordinates.

If demands are limited, the piece sets have to be considered. Let  $B_0 = (d_1, d_2, \dots, d_n)$  be the set with full demands  $d_i$ , and  $B$  a generic subset with frequencies  $b_i \leq d_i$ .

Let  $f_C(l, w, B)$  be the best profit for a rectangle  $(l, w)$  with the available piece set  $B$ . The following recursion (3)–(4.a, 4.b, 4.c), similar to (1)–(2.a, 2.b, 2.c),

allows to compute all  $f_C(l, w, B)$  values. In (4.b) and (4.c) the set  $B$  is partitioned between the two sub-rectangles generated by the vertical or the horizontal cut respectively.

$$f_C(l, w, B) = \max\{f_{C,0}(l, w, B), f_{C,v}(l, w, B), f_{C,h}(l, w, B)\}, \quad (3)$$

$$0 \leq l \leq L; 0 \leq w \leq W; \phi \subseteq B \subseteq B_0$$

$$f_{C,0}(l, w, B) = \max\{0; \pi_i: l_i \leq l, w_i \leq w, b_i \geq 1\} \quad (4.a)$$

$$f_{C,v}(l, w, B) = \max\{0; f_C(x_1, w, B_1) + f_C(x_2, w, B - B_1):$$

$$0 < x_1 \leq x_2, x_1 + x_2 = l, \phi \subseteq B_1 \subseteq B\} \quad (4.b)$$

$$f_{C,h}(l, w, B) = \max\{0; f_C(l, y_1, B_1) + f_C(l, y_2, B - B_1):$$

$$0 < y_1 \leq y_2, y_1 + y_2 = w, \phi \subseteq B_1 \subseteq B\} \quad (4.c)$$

Dynamic programming on (3)–(4.a, 4.b, 4.c) is impracticable for the huge number of subsets. In relation (3) any frequency  $b_i$  can be reduced if bigger than  $\lfloor (l \cdot w) / (l_i \cdot w_i) \rfloor$ , as noticed in [9], but the ratio  $\lfloor l / l_i \rfloor \cdot \lfloor w / w_i \rfloor$  is smaller and still correct as limit.

In [11] there is a top-down implementation of (3)–(4.a, 4.b, 4.c). The authors define the sum of two cutting patterns  $B' = (b'_1, b'_2, \dots, b'_n)$  and  $B'' = (b''_1, b''_2, \dots, b''_n)$  as the pattern with frequencies  $\min\{d_i, b'_i + b''_i\}$ . They assign a set  $F(l, w)$  of best partial patterns to each couple  $(l, w)$ . In the first inner loop, for example, for each  $x_1$  they add to  $F(l, w)$  all patterns from  $F(x_1, w) \oplus F(l - x_1, w)$ , defined as  $\{B_1 + B_2: B_1 \in F(x_1, w), B_2 \in F(l - x_1, w)\}$ , and then keep only the best patterns of  $F(l, w)$ .

In [9] and [27] a different recursion is exploited, in a relaxed state space. Any piece set  $B = \{b_1, b_2, \dots, b_n\}$  is mapped to a number  $s(B)$  through the scalar product  $B \times S$ . The mapping  $S = \{s_1, s_2, \dots, s_n\}$  has pre-fixed non-negative integers. The advantage is in the reduced size of the mapped space. An upper bound function  $f_S$  is then obtained with the recursion (5)–(6.a, 6.b, 6.c), i.e.  $f_C(l, w, B) \leq f_S(l, w, s(B))$  is valid.

$$f_S(l, w, s) = \max\{f_{S,0}(l, w, s), f_{S,v}(l, w, s), f_{S,h}(l, w, s)\}, \quad (5)$$

$$0 \leq l \leq L; 0 \leq w \leq W; 0 \leq s \leq s_0 = B_0 \times S$$

$$f_{S,0}(l, w, s) = \max\{0; \pi_i: l_i \leq l, w_i \leq w, s_i \leq s\} \quad (6.a)$$

$$f_{S,v}(l, w, s) = \max\{0; f_S(x_1, w, s_1) + f_S(x_2, w, s - s_1): 0 < x_1 \leq x_2,$$

$$x_1 + x_2 = l, 0 \leq s_1 \leq s\} \quad (6.b)$$

$$f_{S,h}(l, w, s) = \max\{0; f_S(l, y_1, s_2) + f_S(l, y_2, s - s_2): 0 < y_1 \leq y_2,$$

$$y_1 + y_2 = w, 0 \leq s_2 \leq s\} \quad (6.c)$$

## 1.2 Branch and Bound Methods

The first type of tree search is based on the *top-down* approach [8]. The root node contains the sheet as *open* rectangle. When a node  $N$  is chosen, its branching consists in selecting an open rectangle from  $N$ , and then a child node is generated for any possible guillotine cut to apply only on it. A special *0-cut* makes the rectangle *closed*, i.e. only one piece can be extracted. Hence, any node  $N$  contains the rectangles generated by the sequence of guillotine cuts added from the root to  $N$ .

At any node, if  $(\alpha_1, \beta_1)$ ,  $(\alpha_2, \beta_2)$ ,  $\dots$ ,  $(\alpha_k, \beta_k)$  are its open rectangles, an upper bound is computed by using the knapsack function but, for the set  $C$  of closed ones, the best matching pieces-rectangles is found, with value  $opt(C)$ . The result is

$$f_U(\alpha_1, \beta_1) + f_U(\alpha_2, \beta_2) + \dots + f_U(\alpha_k, \beta_k) + opt(C) \quad (7)$$

The first method with a *bottom-up* approach was heuristic [31, 33], later improved into an exact tree search by [32]. It merges the rectangles (*builds*) with a separating horizontal or vertical guillotine cut, starting by the pieces up to the sheet. The root node is a dummy node with an empty build. At the first level,  $n$  different nodes contain a rectangle equivalent to one piece of the corresponding type. Hence, the search starts with  $n$  non-branched (or *open*) nodes, and their set is denoted by  $O$ . Any node is identified with its build, i.e. a rectangle and a cutting pattern, and we use the symbol  $B$  also to denote the builds and related nodes.

At any step, a build  $B$  is chosen from  $O$ , in order to be branched, hence moved into the set  $C$  of *closed* nodes. The chosen build is combined with all builds from  $C$  (itself too) in two ways, i.e. vertical or horizontal. Each combination generates a new build, placed into  $O$  according to two conditions, otherwise it is dropped. First, it has to fit inside the sheet. Second, no demand has to be exceeded.

## 2 Upper Bounds Classification

We present four criteria to characterize the upper bounds in tree searches for C2DC, describing some details from topic literature papers. **(a) the type of relaxation.** Four types can be identified: unconstrained relaxation, state space relaxation, one-dimensional (or geometric) relaxation, non-guillotine relaxation. In geometric case, additional options are the continuous frequencies or the container relaxation. We avoid treating ILP relaxations, since just two ILP models exist, which allow to solve only small instances [14, 25]. **(b) the kind of node** for which the bound is computed: root or inner. In the first case the full sheet is available. In the second case there is a residual area, with non-rectangular shape for bottom-up, or separated parts for top-down. **(c) the available piece set.** Some upper bounds are pre-computed at the begin of the algorithm with all desired pieces, and we call them *free*. Other upper bounds are frequently computed using a residual subset of pieces.

We refer to them as *residual*. **(d) the geometric compatibility** of piece types and demands with respect to available areas. We shall describe the options in combination to the ones selected for the three previous features.

The four criteria are not independent. For example, in the root node all upper bounds are free, the unconstrained relaxation implicitly integrates the geometric compatibility, the geometric relaxations imply the non-guillotine relaxation.

## 2.1 Geometric Relaxations

The first two relaxations have been presented in Sect. 1.1, with the knapsack function (unconstrained relaxation) and the state space relaxation. Two types of one-dimensional relaxation exist. We call the first *1D-area*, since it considers only the area of both sheet and pieces [17, 20], by solving a knapsack problem. In the second, the pieces are split into slices [36, 37], hence we call it *1D-slice*.

The slices are used to fill the available area, which can be broken into empty strips. The computation is simplified since only one pattern is used for equivalent strips. In particular, for any piece type  $t_i$ , the horizontal slice has length  $l'_{i,h} = l_i$ , width  $w'_{i,h} = 1$ , profit  $\pi'_{i,h} = \pi_i / w_i$  and demand  $d'_{i,h} = \min\{d_i, \lfloor L/l_i \rfloor\}$ . The slice profits are fractional, hence a truncation  $\lfloor \cdot \rfloor$  is applied on the overall bound, even when not explicit. This happens whenever continuous piece frequencies are used.

The best pattern is computed through a knapsack problem with capacity  $L$ . It is vertically replicated  $W$  times with no care about the limited demands  $d_i$ , hence in this phase the unconstrained relaxation is added. Vertical slices are similarly defined and used, and the minimum is selected between the two orientations.

The two relaxations do not dominate each other, since the 1D-slice considers the geometric limits on each axis, but it uses fractional frequencies and a partial unconstrained relaxation, which would be full if  $d'_{i,h}$  is replaced by  $d''_{i,h} = \lfloor L/l_i \rfloor$ . Also the 1D-area can be integrated with these relaxations on the frequencies.

### 2.1.1 Non-guillotine Relaxations

Geometric relaxations implicitly include the drop of the guillotine constraint, since it has no meaning when treating with one-dimensional pieces. However the non-guillotine relaxation can be independently considered and a better upper bound would be obtained by dropping only this constraint. Unfortunately the problem becomes more difficult. This kind of bounds is explicitly mentioned in [5, 11]. The former is addressed to the U2DC. In the latter the non-guillotine solutions from literature are manually imported, but only for the full sheet.

In literature two groups of upper bounds can be identified for the non-guillotine case, which turn out to be valid also for the guillotine case. The first group is based on geometric relaxations extending the 1D-slice and the 1D-area, often from proposals given in the context of different problems, as the Strip Packing and the

Orthogonal Packing problems. In 1D-slice case, one-dimensional cutting stock problems raise (with strips as piece containers), for which ILP models exist in literature. In 1D-area case, piece dimensions are modified through smart strategies (e.g. *dual feasible functions*), leading to greater areas. However, all corresponding bounds have been mainly experienced to check if a set of pieces fits into a rectangle. Because of this different perspective, we shall not discuss further details, in particular about ILP models and relaxations, and we just underline that area increasing represents a powerful pre-processing but it does not affect the skeleton of our classification. The interested reader is addressed to [1, 3, 4, 13, 24].

The second group comes from relaxations (constraint removal, lagrangian, etc.) applied to ILP models of the full non-guillotine problem, but we pursue the choice to avoid using ILP models. The interested reader is addressed to [2, 6, 7].

## 2.2 Root Node and Inner Nodes

In the root node, the full sheet is considered, with no difference between top-down and bottom-up approaches, as for the upper bounds previously described.

In top-down case, for the inner nodes the model of upper bounds has been given in (7). In [20], for each open rectangle, the knapsack function and the 1D-area relaxation are used, selecting the minimum between the two corresponding sums.

In the bottom-up methods, for any inner node with a build  $B$ , an upper bound is given by  $\pi(B) + u(\Gamma)$ , where  $u(\Gamma)$  is a bound for the best way to use the complementary residual surface  $\Gamma$  when  $B$  is placed on the bottom-left corner of the sheet.

No state space relaxation has been proposed for  $u(\Gamma)$ . An upper bound with 1D-area relaxation can be computed straightly by using the area of  $\Gamma$ . For the 1D-slice relaxation,  $\Gamma$  can be divided in two parts [38] through a horizontal line along the top edge of  $B$ . The bottom-right part of  $\Gamma$  is filled by using shorter strips.

In [32], an upper bound with unconstrained relaxation is based on the backward recursion (8.a, 8.b)–(9.a, 9.b), where  $f_U$  is the knapsack function and  $B$  has dimensions  $(l, w)$ .

$$V_U(L, W) = 0 \tag{8.a}$$

$$V_U(l, w) = \max\{v_U(l, w), h_U(l, w)\}, 0 \leq l \leq L; 0 \leq w \leq W \tag{8.b}$$

$$v_U(l, w) = \max\{0; V_U(l+x, w) + f_U(x, w): 0 < x \leq L-l\} \tag{9.a}$$

$$h_U(l, w) = \max\{0; V_U(l, w+y) + f_U(l, y): 0 < y \leq W-w\} \tag{9.b}$$

The value  $V_U(l, w)$  plays the role of  $u(\Gamma)$ . Another bound from [32], dominated by  $V_U(l, w)$  but faster, is given by  $f_U(L-l, W) + f_U(L, W-w)$ .

### 2.3 Free and Residual Upper Bounds

All upper bounds described above are *free* (all pieces available). In the inner nodes, *residual* upper bounds can be used, by taking into account only the residual available pieces. For unconstrained and state space relaxations there are no literature examples. For the 1D-area the bounds are based on reduced frequency limits.

For the 1D-slice relaxation, at the begin of the bottom-up algorithm in [36], the authors compute the best strips not only with the full piece set  $B_0$ , but also with the  $n$  subsets where (only) one type is removed. In any node, also the bounds associated to types with no residual demands are selected, and the minimum is chosen.

In general, note that the free upper bounds are pre-computed at the begin of the algorithm, hence there is no need to be faster by using continuous frequencies, which are instead useful for the residual upper bounds, frequently computed.

### 2.4 Geometric Compatibility

In this section we present residual and free upper bounds from literature, based on one-dimensional relaxations, with focus on the options to filter piece types and piece demands, according to the geometric compatibility with respect to the available areas. Let  $T(\alpha, \beta)$  be the subset containing the type indexes  $i$  for which  $l_i \leq \alpha$  and  $w_i \leq \beta$ . We say that these pieces are *compatible* with the rectangle  $(\alpha, \beta)$ .

For unconstrained and state space relaxations, the restriction to the subsets  $T(\alpha, \beta)$  is implicit, since the constraints  $l_i \leq l$  and  $w_i \leq w$  are used in (1) and in (6.a).

We shall first discuss the compatibility for residual 1D-area upper bounds, then for strips construction, and finally for some free 1D-area upper bounds.

#### 2.4.1 Residual Upper Bounds with Compatibility

Two residual upper bounds related to the 1D-area relaxation exist, both for the bottom-up case. The first appeared in [10] with the sum of all residual piece profits, with no limits on the sum of their area, hence ignoring  $\Gamma$ , and we refer to that as *container relaxation*. However, the authors only use the piece types compatible with  $\Gamma$ . We denote this set by  $T_\Gamma$ . Hence, if  $b_i$  are the frequencies in  $B$ ,  $u(\Gamma)$  is

$$\sum_{i \in T_\Gamma} (d_i - b_i) \cdot \pi_i \quad (10)$$

In [32], if this expression is zero, then it is used in place of  $V_U(l, w)$ .

For the second upper bound, continuous frequencies are used, but it is better that (10) since it takes into account the area of  $\Gamma$  as upper limit [18, 36]. For a given build  $B$ , the knapsack formulation with  $b'_i$  as variable frequencies is



$$\max \sum_{i \in T_\Gamma} b'_i \cdot \pi_i \quad \text{s.t. } 0 \leq b'_i \leq d_i - b_i, \sum_{i \in T_\Gamma} b'_i \cdot l_i \cdot w_i \leq A = \text{area}(\Gamma) \quad (11)$$

### 2.4.2 Strips with Compatible Pieces

A bound for 1D-slice relaxation originally appeared in a top-down approach for U2DC [37]. The best horizontal strip compatible with a rectangle  $(\alpha, \beta)$  is

$$u_h(\alpha, \beta) = \max \sum_{i \in T} \pi'_{i,h} \cdot b_i \quad \text{s.t. } \sum_{i \in T} l_i \cdot b_i \leq \alpha; b_i \text{ integer}; b_i \geq 0 \quad (12)$$

The authors compute (12) only for  $\beta$  equal to piece widths. A similar definition  $u_v(\alpha, \beta)$  is valid for the vertical slices. The minimum is selected as upper bound.

In [21, 38] this bound is applied to the bottom-up (for U2DC), but it is just computed for the full sheet, and  $u_h(\alpha, \beta)$  is replaced by  $u_h(\alpha)$ , with  $\beta = W$  in (12).

The bound was adapted to C2DC in [36] by adding constraints  $b_i \leq d_i$  in (12). We refer to these strips as *constrained strips*. For a build with size  $(l, w)$ ,  $u(\Gamma)$  is given, for example with horizontal strips, by (13), which we refine later according to the discrete points. The authors also select the best residual bound (Sect. 2.3).

$$\lfloor u_h(L-l) \cdot w + u_h(L) \cdot (W-w) \rfloor \quad (13)$$

### 2.4.3 Free 1D-Area Upper Bounds

In the top-down approach of [20] the geometric compatibility is integrated to the 1D-area relaxation. For an open  $(\alpha, \beta)$ , an integer knapsack problem is solved:

$$\max \sum_{i \in T} b_i \cdot \pi_i \quad \text{s.t. } \sum_{i \in T} b_i \cdot l_i \cdot w_i \leq A = \alpha \cdot \beta; b_i \text{ integer}; 0 \leq b_i \leq c_i(\alpha, \beta) \quad (14)$$

with capacity  $\alpha \cdot \beta$ , the piece types of  $T(\alpha, \beta)$ , and the upper limits  $c_i(\alpha, \beta) = \min\{d_i, \lfloor \alpha/l_i \rfloor \cdot \lfloor \beta/w_i \rfloor\}$  for the integer variable frequencies  $b_i$ . We refer to  $c_i$  as *specific demands*. When they are used we say there is *full* geometric compatibility.

For the bottom-up approach, the formulation given in [17] is equivalent to (14), but the constraint  $l_i \leq \alpha$  and  $w_i \leq \beta$  should be replaced by  $l_i \leq L-\alpha$  or  $w_i \leq W-\beta$ , as noticed also in [36]. In [18] an upper limit is defined for frequencies. We denote it by  $c_{\Gamma,i}(\alpha, \beta)$ . The authors set  $c_{\Gamma,i}(\alpha, \beta) = 0$  if  $l_i > L-\alpha$  and  $w_i > W-\beta$ . When only the first condition is valid, they set  $c_{\Gamma,i}(\alpha, \beta) = \min\{d_i, \lfloor L \cdot (W-\beta) / (l_i \cdot w_i) \rfloor\}$ , and similarly if only the second is satisfied. In all other cases, the total area of  $\Gamma$  is exploited, by setting  $c_{\Gamma,i}(\alpha, \beta) = \min\{d_i, \lfloor \text{area}(\Gamma) / (l_i \cdot w_i) \rfloor\}$ .

## 2.5 Mixed Upper Bounds

Upper bounds can be mixed by selecting the minimum. A better strategy, related to (1)–(2.a, 2.b, 2.c) and (8.a, 8.b)–(9.a, 9.b), integrates the unconstrained and the 1D-area relaxations. In [23],  $f_U(l, w)$  is replaced by  $f_{U,m}(l, w) = \min\{f_{1D}(l, w), f_U(l, w)\}$ , where  $f_{1D}$  refers to a 1D-area relaxation. Hence,  $f_{U,m}$  can be smaller than  $f_U$ . The authors propagate any smaller value also to the bigger rectangles, since they use  $f_{U,m}$  in (1.b, 1.c) instead of  $f_U$ , so leading to reduced functions  $f_{U,h,m}$  and  $f_{U,v,m}$ .

A similar strategy appeared in [18], with different perspectives. First, the authors applied the mix to (8.a, 8.b)–(9.a, 9.b). Second, it is performed by replacing  $f_U(x, y)$  with  $\min\{f_{1D}(x, y), f_U(x, y)\}$  in (9.a, 9.b). Hence, this mix only reduces the propagation of  $f_U$ .

## 2.6 Discrete Points

A way to improve a tree search approach is with the discretization of the cut coordinates by considering only the integer combinations of the piece dimensions [8, 16]. We denote the *discrete points* sets by  $D_X$  (lengths) and  $D_Y$  (widths).

The useful portion of a rectangle  $(l, w)$  has dimensions  $(\langle l \rangle_x, \langle w \rangle_y)$  and area  $\langle l \rangle_x \cdot \langle w \rangle_y$ , where  $\langle l \rangle_x = \max\{x \in D_X: x \leq l\}$  and  $\langle w \rangle_y = \max\{y \in D_Y: y \leq w\}$ . These reductions imply smaller upper bounds for the 1D-slice and 1D-area relaxations.

No example exists for top-down methods. In the bottom-up case, the dimensions  $l$  and  $w$  of any build  $B$  are necessarily discrete points. The dimensions  $L-l$  and  $W-w$  of the complementary surface  $\Gamma$  are reduced to  $\langle L-l \rangle_x$  and  $\langle W-w \rangle_y$  in [35]. We denote the reduced surface by  $\langle \Gamma \rangle_d$ . A reduction of  $\Gamma$  through discrete points is also described in [18], before defining  $c_{\Gamma,i}$ , but no formulation was given.

For the 1D-slice relaxation, the reduction has no effect on the construction of the best strips but, for example, the replication of the horizontal ones is reduced on the  $y$ -axis. Hence, the following refined expression is used in [36] instead of (13)

$$[u_h(L-l) \cdot (W - \langle W-w \rangle_y) + u_h(L) \cdot \langle W-w \rangle_y] \quad (15)$$

where, in the second term, the factor  $(W-w)$  is replaced by  $\langle W-w \rangle_y$  whereas, in the first term, the factor  $w$  has been increased with an equivalent gap.

## 3 Literature Categorization

We characterize the upper bounds used in a set of significant literature papers, by using our four criteria. Tables 1 and 2 refer to the top-down and bottom-up case respectively. Last column reports if a bound mixing is used. In some cases the mix

**Table 1** Literature upper bounds for top-down algorithms

Paper	Relaxation	Root/Inner	Free/Residual	Geometric compatibility	Bound mixing
Christofides and Whitlock [8]	Unconstrained	Both	Free	Compatible	No
Christofides and Hadjiconstantinou [9]	State space	Both	Free	Compatible	No
Morabito and Arenales [26]	1D-Area + continuous	Both	Residual	Compatible	No
Hifi and Zissimopoulos [20]	1D-area	Both	Free	Full	Yes
Hifi [19]	1D-area + continuous	Inner	Free	Compatible	No
Morabito and Pura [27]	State space	Both	free	Compatible	No

**Table 2** Literature upper bounds for bottom-up algorithms

Paper	Relaxations	Root/Inner	Free/Residual	Geometric compatibility	Bound mixing
Wang [33]	1D-area + continuous + unconstrained	Both	Free	Unlimited	No
Viswanathan and Bagchi [32]	Unconstrained	Both	Free	Compatible	Yes
Tschöke and Holthöfer [30]	1D-area	Both	Free	Unlimited	Yes
Hifi [17]	1D-area	Both	Free	Unlimited	Yes
Cung et al. [10]	Container	Inner	Residual	Compatible	Yes
León et al. [23]	Unconstrained mixed to 1D-area	Both	Free	Unlimited on 1d-area	No
Hifi et al. [18]	1D-area	Root	Free	Unlimited	Yes
	Unconstrained mixed to 1D-area	Both	Free	Compatible on $f_u$	
	1D-area + continuous	Both	Free	Compatible	
Dolatabadi et al. [11]	1D-area	Inner	Residual	Unlimited	Yes
Yoon et al. [36]	1D-slice with discrete points	Both	Free/residual	Unlimited	Yes
	1D-area + continuous	Inner	Residual	Compatible	
Wei and Lim [35]	1D-area	Root	Free	Unlimited	Yes
	1D-area	Inner	Free	Unlimited	
	1D-area + continuous	Inner	Residual	Unlimited	

involves bounds from other papers, as better detailed in an extended report on-line, accessible at [opslab.dieti.unina.it](http://opslab.dieti.unina.it) and containing also formal formulations.

## References

1. Alvarez-Valdes, R., Parreño, F., Tamarit, J.M.: A branch and bound algorithm for the strip packing problem. *OR Spect.* **31**, 431–459 (2009)
2. Baldacci, R., Boschetti, M.A.: A cutting-plane approach for the two-dimensional orthogonal non-guillotine cutting problem. *Eur. J. Oper. Res.* **183**, 1136–1149 (2007)
3. Belov, G., Kartak, V.M., Rohling, H., Scheithauer, G.: One-dimensional relaxations and LP bounds for orthogonal packing. *Int. Trans. Oper. Res.* **16**, 745–766 (2009)
4. Belov, G., Kartak, V.M., Rohling, H., Scheithauer, G.: Conservative scales in packing problems. *OR Spect.* **35**, 505–541 (2013)
5. Birgin, E., Lobato, R., Morabito, R.: Generating unconstrained two-dimensional non-guillotine cutting patterns by a recursive partitioning algorithm. *J. Oper. Res. Soc.* **63**, 183–200 (2012)
6. Boschetti, M.A., Mingozzi, A., Hadjiconstantinou, E.: New upper bounds for the two-dimensional orthogonal non-guillotine cutting stock problem. *IMA J. Man. Math.* **13**, 95–119 (2002)
7. Caprara, A., Monaci, M.: On the two-dimensional knapsack problem. *Oper. Res. Lett.* **32**, 5–14 (2004)
8. Christofides, N., Whitlock, C.: An algorithm for two-dimensional cutting problems. *Oper. Res.* **25**, 30–44 (1977)
9. Christofides, N., Hadjiconstantinou, E.: An exact algorithm for orthogonal 2-D cutting problems using guillotine cuts. *Eur. J. Oper. Res.* **83**, 21–38 (1995)
10. Cung, V., Hifi, M., Le Cun, B.: Constrained two-dimensional cutting stock problems a best-first branch-and-bound algorithm. *Int. Trans. Oper. Res.* **7**, 185–210 (2000)
11. Dolatabadi, M., Lodi, A., Monaci, M.: Exact algorithms for the two-dimensional guillotine knapsack. *Comp. Oper. Res.* **39**, 48–53 (2012)
12. Fayard, D., Hifi, M., Zissimopoulos, V.: An efficient approach for large-scale two-dimensional guillotine cutting stock problems. *J. Oper. Res. Soc.* **49**, 1270–1277 (1998)
13. Fekete, S.P., Schepers, J.: A general framework for bounds for higher-dimensional orthogonal packing problems. *Math. Meth. Oper. Res.* **60**, 311–329 (2004)
14. Furini, F., Malaguti, E., Thomopulos, D.: Modeling two-dimensional guillotine cutting problems via integer programming (2014). <http://www.optimization-online.org>
15. Gilmore, P., Gomory, R.: The theory and computation of knapsack functions. *Oper. Res.* **14**, 1045–1074 (1966)
16. Herz, J.: Recursive computation procedure for two-dimensional stock cutting. *IBM J. Res. Dev.* **16**, 462–469 (1972)
17. Hifi, M.: An improvement of Viswanathan and Bagchi's exact algorithm for constrained two-dimensional cutting stock. *Comp. Oper. Res.* **24**, 727–736 (1997)
18. Hifi, M., M'Hallah, R., Saadi, T.: Approximate and exact algorithms for the double-constrained two-dimensional guillotine cutting stock problem. *Comp. Opt. App.* **42**, 303–326 (2009)
19. Hifi, M.: Dynamic programming and hill-climbing techniques for constrained two-dimensional cutting stock problems. *J. Combinat. Optim.* **8**, 65–84 (2004)
20. Hifi, M., Zissimopoulos, V.: Constrained two-dimensional cutting: an improvement of Christofides and Whitlock's exact algorithm. *J. Oper. Res. Soc.* **48**, 324–331 (1997)
21. Kang, M., Yoon, K.: An improved best-first branch-and-bound algorithm for unconstrained two-dimensional cutting problems. *Int. J. of Prod. Res.* **49**, 4437–4455 (2011)

22. Kellerer, H., Pferschy, U., Pisinger, D.: *Knapsack Problems*, Springer (2004)
23. León, C., Miranda, G., Rodríguez, C., Segura C.: 2D cutting stock problem: A new parallel algorithm and bounds. *Euro. Conf. Parallel Process.* 795–804 (2007)
24. Martello, S., Monaci, M., Vigo, D.: An exact approach to the strip-packing problem. *INFORMS J. Comp.* **3**, 310–319 (2003)
25. Messaoud, S.B., Chu, C., Espinouse, M.: Characterization and modelling of guillotine constraints. *Eur. J. Oper. Res.* **191**, 112–126 (2008)
26. Morabito, R., Arenales, M.: Staged and constrained two-dimensional guillotine cutting problems: an AND/OR-graph approach. *Eur. J. Oper. Res.* **94**, 548–560 (1996)
27. Morabito, R., Pureza, V.: A heuristic approach based on dynamic programming and and/or-graph search for the constrained two-dimensional guillotine cutting problem. *Ann. Oper. Res.* **179**, 297–315 (2010)
28. Russo, M., Sforza, A., Sterle, C.: An improvement of the knapsack function based algorithm of Gilmore and Gomory for the unconstrained two-dimensional guillotine cutting problem. *Int. J. Prod. Econ.* **145**, 451–461 (2013)
29. Russo, M., Sforza, A., Sterle, C.: An exact dynamic programming algorithm for large-scale unconstrained two-dimensional guillotine cutting problems. *Comp. Oper. Res.* **50**, 97–114 (2014)
30. Tschöke, S., Holthöfer, N.: A new parallel approach to the constrained two-dimensional cutting stock problem. *Parallel Algorithms for Irregularly Structured Problems*, pp. 285–300. Springer (1995)
31. Vasko, F.J., Bartkowski, C.L.: Using Wang's two-dimensional cutting stock algorithm to optimally solve difficult problems. *Int. Trans. Oper. Res.* **16**, 829–838 (2009)
32. Viswanathan, K., Bagchi, A.: Best-first search methods for constrained two-dimensional cutting stock problems. *Oper. Res.* **41**, 768–776 (1993)
33. Wang, P.Y.: Two algorithms for constrained two-dimensional cutting stock problems. *Oper. Res.* **31**, 573–586 (1983)
34. Wäscher, G., Haußner, H., Schumann, H.: An improved typology of cutting and packing problems. *Eur. J. Oper. Res.* **183**, 1109–1130 (2007)
35. Wei, L., Lim, A.: A bidirectional building approach for the 2D constrained guillotine knapsack packing problem. *Eur. J. of Oper. Res.* **242**, 63–71 (2015)
36. Yoon, K., Ahn, S., Kang, M.: An improved best-first branch-and-bound algorithm for constrained two-dimensional guillotine cutting problems. *Int. J. Prod. Econ.* **51**, 1608–1692 (2013)
37. Young-Gun, G., Kang, M.: A new upper bound for unconstrained two-dimensional cutting and packing. *J. Oper. Res. Soc.* **53**, 587–591 (2002)
38. Young-Gun, G., Seong, Y.J., Kang, M.K.: A best-first branch and bound algorithm for unconstrained two-dimensional cutting problems. *Oper. Res. Lett.* **31**, 301–307 (2003)

**Part X**  
**Railway and Maritime Optimization**

# Some Complexity Results for the Minimum Blocking Items Problem

Tiziano Bacci, Sara Mattia and Paolo Ventura

**Abstract** In this paper, we study the *Minimum Blocking Items Problem* (MBIP) as a generalization of the Bounded Coloring Problem for Permutation Graphs and we motivate our interest by discussing some practical applications of MBIP to the context of minimizing reshuffle operations in a container yard. Then we present some results on the computational complexity of MBIP.

**Keywords** Bounded coloring problem · Block relocation problem · Complexity

## 1 Introduction

Let  $\mathcal{S}$  be a system defined by  $w$  stacks of capacity (in terms of available slots)  $h$ . Then let  $\{1, \dots, n\}$  be a set of items that enter the system according to the order defined by the vector  $\phi$ , where  $\phi_i$  is the  $i$ -th entering item.

The stacks can store items according to a last-in/first-out policy. When an item enters the system, it has to be allocated in one of the stacks, in the first slot available from the bottom to the top of the stack.

We call *configuration*, and we denote it by  $M$ , an assignment of the items  $\{1, \dots, n\}$  to the slots of the  $w$  stacks that is compatible with the entering order  $\phi$ . Let  $M(j, k)$  denote the item allocated in the  $k$ -th position of stack  $j$  ( $M(j, k) = 0$  if the slot is empty). Then, to be coherent with the order  $\phi$ , for each couple  $i, i' \in \{1, \dots, n\}$  with  $i < i'$ , it cannot exist a stack  $j$  such that  $M(j, k) = \phi_{i'}$  and  $M(j, k + 1) = \phi_i$  for some  $1 \leq k < h$ .

---

T. Bacci (✉) · S. Mattia · P. Ventura  
Istituto di Analisi dei Sistemi ed Informatica (IASI), Consiglio Nazionale  
delle Ricerche (CNR), via dei Taurini 19, 00185 Rome, Italy  
e-mail: tiziano.bacci@iasi.cnr.it

S. Mattia  
e-mail: sara.mattia@iasi.cnr.it

P. Ventura  
e-mail: paolo.ventura@iasi.cnr.it

$M_a$			$M_b$		
0	0	7	0	0	4
4	2	5	7	2	5
3	1	6	1	3	6

**Fig. 1** Solutions  $M_a$  and  $M_b$  for Example 1

Furthermore, we say that item  $i = M(j, k)$  is *blocking* if there is an item  $i' = M(j, k') < i$  for some  $1 \leq k' < k$  (and therefore  $i'$  is said to be *blocked*). Given a configuration  $M$ , we denote by  $BI(M)$  the total amount of blocking items of  $M$ . Given an *input instance* defined by  $n, \phi, w$ , and  $h$ , the *Minimum Blocking Items Problem* (MBIP) is to find the configuration  $M$  that minimizes  $BI(M)$ .

Below we provide an example of MBIP, with feasible and optimal solutions.

*Example 1* Consider the input instance defined by  $n = \{1, \dots, 7\}$ ,  $w = 3$ ,  $h = 3$ , and  $\phi = [1, 3, 6, 5, 7, 4, 2]$ . Figure 1 shows two solutions  $M_a$  and  $M_b$  corresponding to different values of  $BI(M)$ . Specifically  $BI(M_a) = 3$  and  $BI(M_b) = 1$ . An empty slot is denoted by 0. In both cases, the blocking items are colored in gray. It is not difficult to see that  $M_b$  is, in fact, the optimal solution of the problem.

In the following, we show that the problem of checking if there exists a configuration  $M$  such that  $BI(M) = 0$  is equivalent to the *Bounded Coloring Problem on Permutation Graphs* [1], denoted here by BCPPG. Given a graph  $G(V, E)$ , a set of colors  $\{c_1, \dots, c_w\}$  and a positive value  $h$ , the Bounded Coloring Problem consists of finding an assignment of colors to the nodes of the graph so that the endpoints of each edge have different colors and color  $c_j$  is assigned to at most  $h$  nodes. We call such an assignment a *feasible coloring* of  $G$ . Given a permutation  $\phi$  of items  $\{1, \dots, n\}$ , the corresponding *permutation graph*  $G(\phi) = (V(\phi), E(\phi))$  is defined as:  $V(\phi) = \{1, \dots, n\}$ ;  $\{i, j\} \in E(\phi)$  if and only if  $(i - j)(\phi_i - \phi_j) > 0$ .

Now, if we associate nodes with items and colors with stacks, it is not difficult to see that BCPPG has a positive answer (i.e.  $G$  admits a feasible coloring) if and only if the corresponding MBIP admits a configuration  $M$  such that  $BI(M) = 0$ . Therefore the following holds true:

**Observation 1** *The Minimum Blocking Items Problem is a generalization of the Bounded Coloring Problem on Permutation Graphs.*

Two other optimization problems that are generalizations of the BCPPG can be defined by minimizing the number of edges (vertices, resp.) that have to be removed from  $G$  in order to obtain a graph that admits a feasible coloring. In this cases, we talk of *Edge Deletion Bounded Coloring Problem on Permutation Graphs* (ED-BCPPG) and *Vertex Deletion Bounded Coloring Problem on Permutation Graphs* (VD-BCPPG), respectively.

Indeed, one could think that MBIP could be reduced to one of these problems. The following two examples show this is not true.



*Example 2* Consider the input instance defined by  $n = \{1, \dots, 8\}$ ,  $w = 2$ ,  $h = 4$ , and  $\phi = [3, 7, 2, 1, 6, 8, 5, 4]$ . It is not difficult to see that the (unique) optimal solution to the ED-BBPPG has value 2 and is the one depicted in Fig. 2a, where the dashed edges are those that have to be removed in order to get the feasible solution that assign color 1 (white in the picture) to vertices  $\{3, 2, 1, 8\}$  and color 2 (gray in the picture) to vertices  $\{7, 6, 5, 4\}$ . In the MBIP context, this corresponds to assign items  $\{3, 2, 1, 8\}$  to stack 1, and items  $\{7, 6, 5, 4\}$  to stack 2, as illustrated in Fig. 2b. This solution has 2 blocking items (gray in the picture) while the MBIP optimal solution (shown in Fig. 2c) has value 1.

*Example 3* In this case, let  $n = \{1, \dots, 8\}$ ,  $w = 2$ ,  $h = 4$  and  $\phi = [3, 8, 5, 4, 2, 1, 7, 6]$ . Also here it is not difficult to see that the (unique) optimal solution to the VD-BBPPG has value 1 and is the one depicted in Fig. 3a, where vertex 3 (drown with a dashed line) is the one to be removed in order so to have a feasible solution that assign color 1 (white in the picture) to vertices  $\{5, 4, 2, 1\}$  and color 2 (gray in the picture) to vertices  $\{8, 7, 6\}$ . In the MBIP context, this corresponds to assign items  $\{5, 4, 2, 1\}$  to stack 1, and items  $\{3, 8, 7, 6\}$  to stack 2, as illustrated in Fig. 3b. This solution has 3 blocking items (gray in the picture) while the MBIP optimal solution (shown in Fig. 3c) has value 2.

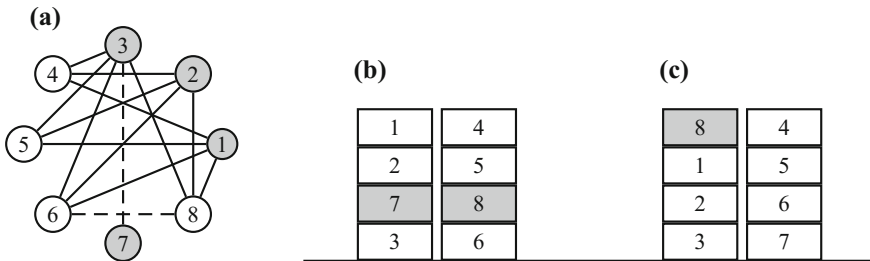


Fig. 2 Picture illustrating Example 2

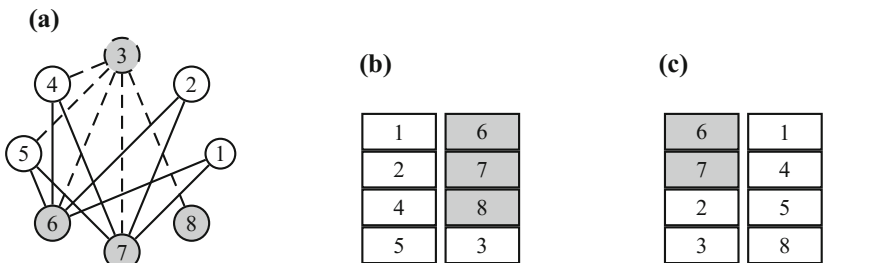


Fig. 3 Illustration for Example 3

## 2 A Practical Application of MBIP

Here we consider the case where the configuration  $M$  defined in the previous section represents a container yard and the items  $\{1, \dots, n\}$  correspond to containers that are piled into  $w$  stacks of height  $h$ . In this context, when a container has to be retrieved from the yard, each container that is located above it has to be reallocated, with a *reshuffle operation* (or, simply, a *reshuffle*), into an other stack of the yard.

If the retrieval order of the containers from the yard is given—in this case we can assume w.l.o.g. that it corresponds to  $[1, 2, \dots, n]$ —then *Block Relocation Problem* (BRP) consists in deciding where to reallocate every container that is moved by a reshuffle operation, in order to minimize the total number of reshuffles needed to retrieve all the items. We denote by  $BR(M)$  such an optimal value.

Reshuffle operations represent the main operational cost in a container yard and, because of the extremely fast growing in the last decades of the number of containers moved worldwide by means of ships, trains, and trucks—and therefore temporary stocked into containers yards—the literature about BRP received a lot of contributions in the recent years.

The BRP is known to be NP-hard [3]. A huge amount of literature has been proposed to define heuristic algorithms and exact approaches (see the recent surveys [2, 7]).

As the blocking items of a yard  $M$  will definitively need a reshuffle operation in order to retrieve the items that are blocked, it is easy to see that  $BR(M) \geq BI(M)$ . Then  $BI(M)$  is used as a lower bound for the optimal solution in the most performing exact approaches for the BRP, as in [4, 9, 11].

Unfortunately, the exact methods proposed so far in the literature can solve in practice only relatively small instances (defined by few dozens items) and cannot tackle real life cases in which thousands of containers are involved. Therefore, heuristic algorithms are very important in practice and most of the procedures proposed in the literature (see [3, 6, 10, 11] among the others) rely on the solution of many *MBIP* (or of some slight variation of it) smaller instances. Indeed, in all these approaches, the common basic idea is to iteratively construct a feasible solution where, at each iteration  $i$  (when item  $i$  is retrieved from the yard), all the containers that have to be reshuffled (ordered from the current top position in the stack to the bottom one) are reallocated in the other stacks of the yard according to a (heuristic or optimal) solution of the associated *MBIP* problem (or some slight modifications of it that vary in the different contributions).

Therefore, we believe that the study of the Minimum Blocking Items Problem, that, to our best knowledge, did not receive much attention in the literature so far, can definitively help to improve the performances of these algorithms.

In the following section, we will give our contribution on some computational complexity aspects of the problem.

**Table 1** Computational complexity of BCPPG

	$h \geq n$	$h < n$ fixed		$h < n$
$w < n$ fixed	<b>P</b>	<b>P</b>	$\Leftarrow$	<b>P</b> [1]
$w < n$	$\uparrow$ <b>P</b> [8]	<b>NP-hard</b> [5]	$\Rightarrow$	<b>NP-hard</b>

### 3 Computational Complexity

It has been proved that BCPPG can be solved in polynomial time when the number  $w$  of colors is fixed [1] or  $h \geq n$  [8]. To the contrary, the problem is known to be hard for any fixed  $6 \leq h < n$  [5]. Such results are reported in Table 1 and arrows represent implications between them. Notice that assuming  $h \geq n$  corresponds to relax the capacity constraints on the stacks. Then BCPPG reduces to a simple coloring problem on permutation graphs, which is easy to solve, as permutation graphs are perfect.

In the following we study the computational complexity of MBIP.

Because of Observation 1, the following holds.

**Lemma 1** *MBIP is NP-hard for  $w < n$  and any fixed  $6 \leq h < n$ .*

As a consequence, we also have

**Corollary 1** *MBIP is NP-hard for  $w < n$  and  $h < n$ .*

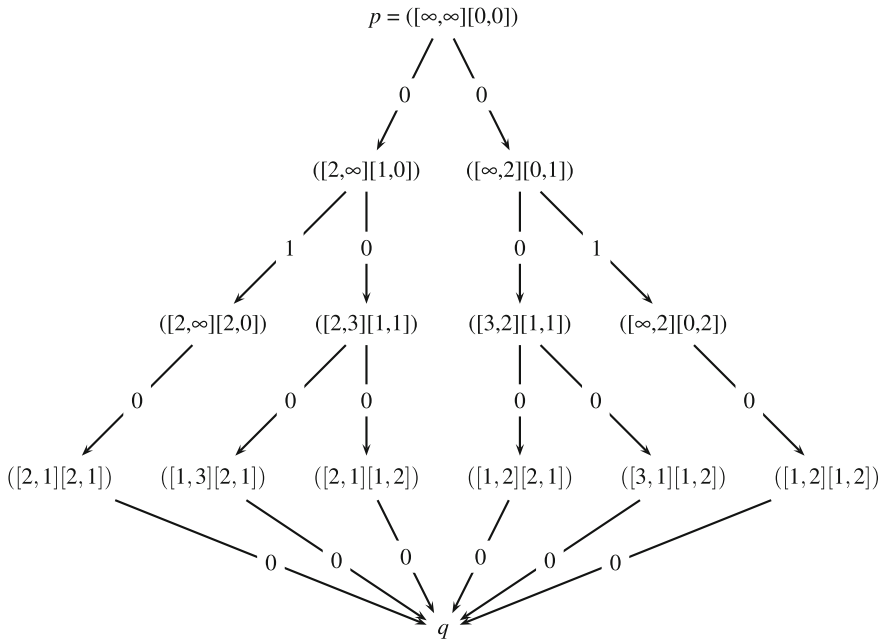
It is not difficult to see that, for  $w \geq n$ , MBIP is easy, independently of  $h$ . Indeed, in this case, it is sufficient to assign every item to a different stack to get a solution of cost 0.

The main contribution of this paper consists of the following two theorems. In particular, we consider the case when  $w < n$  is fixed.

We first prove that MBIP is polynomial time solvable when  $h < n$ .

**Theorem 1** *If  $w < n$  is fixed and  $h < n$ , MBIP is polynomial and can be solved in  $O(n^{w+2}h^w)$ .*

*Proof* We will show that the problem can be reduced to find a shortest path on a suitable oriented graph  $G = (N, A)$  (see Fig. 4). The nodes of  $G$  are associated with the feasible configurations and can be partitioned into layers, where each layer  $0 \leq l \leq n$  contains all (up to stack permutations) the configurations obtained with considering only the items  $\phi_1, \dots, \phi_l$  (at layer 0, the only configuration, named  $p$ , is that with all stacks empty). Each configuration is represented by a couple of  $w$ -dimensional vectors  $s$  and  $t$ , where  $s_i$  is the minimum item located in stack  $i$  ( $s_i = +\infty$  if the stack is empty) and  $t_i$  is the number of items in the stack. Now let  $(s, t)$  be a node in layer  $l < n$ . Then, for each stack  $j$  such that  $t_j < h$ , there is a node  $(s', t')$  in the layer  $l + 1$  that represents the configuration obtained from  $(s, t)$  by locating item  $\phi_{l+1}$  in stack



**Fig. 4** The graph  $G$  of Theorem 1 with  $n = 3, w = 2, h = 2$ , and  $\phi = [2, 3, 1]$

$j$ , and an arc  $a$  from  $(s, t)$  to  $(s', t')$ . Therefore,  $s'_j = \min \{s_j, \phi_{l+1}\}$ ,  $t'_j = t_j + 1$ , and  $s'_{j'} = s_{j'}$ ,  $t'_{j'} = t_{j'}$  for all  $j' \neq j$ . Moreover, the arc  $a$  has weight 1 if  $\phi_{l+1}$  is blocking in  $(s', t')$  (i.e.  $s'_j = s_j$ ) and 0 otherwise ( $s'_j = \phi_{l+1} < s_j$ ). Finally, add a dummy node  $q$  and an arc of weight 0 from all the nodes in the layer  $n$  and  $q$ . Therefore, it is not difficult to see the minimum number of blocking items corresponds to the shortest path from  $p$  to  $q$ .

Since  $G$  is acyclic by construction, such a shortest path can be computed in  $O(|A|)$  time. Moreover, it is not difficult to see that the number of nodes at each layer  $1 \leq l \leq n$  is bounded by  $(l + 1)^w(r + 1)^w$ , where  $r = \min \{l, h\}$ . Therefore,  $|V| = O(n^{w+1}h^w)$  and, as every node has at most  $w$  leaving arcs,  $|A| = O(n^{w+2}h^w)$ .  $\square$

As a consequence, we have also that

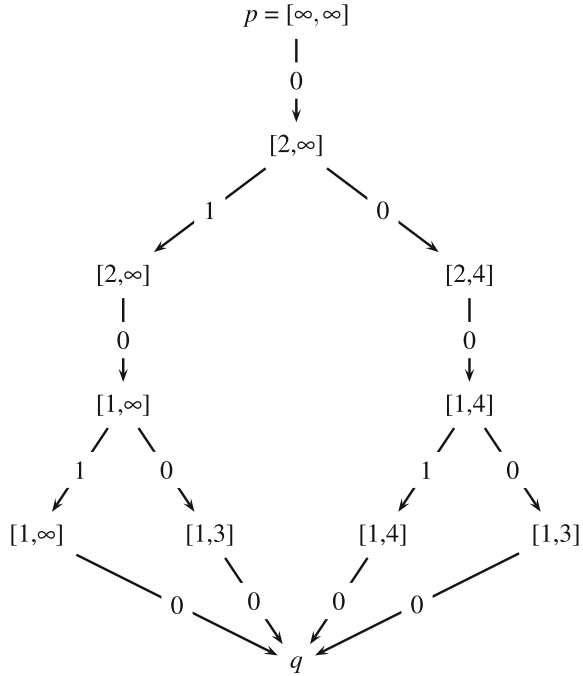
**Corollary 2** *If  $w < n$  and  $h < n$  are fixed, then MBIP can be solved in polynomial time.*

Then we consider the case where the capacity of the stacks is unbounded.

**Theorem 2** *If  $h \geq n$ , MBIP can be solved in  $O(\min \{n^{w+1}, 2^{n+1}\})$  time.*

*Proof* The proof of this theorem mimics the one of Theorem 1 and again we reduce MBIP to a shortest path problem on the acyclic graph  $G$ . In particular, since the

**Fig. 5** Graph  $G$  of Theorem 2 for  $n = 4, w = 2, \phi = [2, 4, 1, 3]$



capacities of the stacks are unbounded, here the feasible configurations can be represented only by the vector  $s$ . Moreover, let  $s$  and  $s'$  be two configurations in the same layer such that  $s_j \geq s'_j$  for each  $j \in \{1, \dots, w\}$ . Then it is not difficult to see that the shortest path from  $s$  to  $q$  cannot be longer than the one from  $s'$  to  $q$ . This implies that, if the shortest path from  $p$  to  $s$  is not longer than to one from  $p$  to  $s'$ , then  $s$  dominates  $s'$  (i.e.  $s'$  can be removed from the graph without effecting the MBIP optimal solution). Therefore, as illustrated in Fig. 5, the outgoing arcs of any configuration  $s$  in layer  $l$  are at most two:  $(s, s^1)$ , of cost 1, associated with the fact that item  $\phi_{l+1}$  is blocking (in this case  $s_j^1 = s_j$  for all  $j$ ), and  $(s, s^0)$ , of cost 0, representing the choice of locating item  $\phi_{l+1}$  in the stack  $\bar{j} = \arg \min \{s_j : s_j > \phi_{l+1}\}$ .

Then, the number of nodes at each layer  $1 \leq l \leq n$  is bounded by  $\min \{(l + 1)^w, 2^l\}$ . Therefore, in this case,  $|V| = O(\min \{n^{w+1}, 2^{n+1}\})$  and, since every node has at most two outgoing arcs, we have that  $|A| = O(\min \{n^{w+1}, 2^{n+1}\})$ .  $\square$

As for any fixed  $w$  there always exists a value  $\bar{n}$  such that  $2^{n+1} > n^{w+1}$  for all  $n > \bar{n}$ , then the following holds

**Corollary 3** *If  $w < n$  is fixed and  $h \geq n$ , MBIP is polynomial and can be solved in  $O(n^{w+1})$ .*

By the previous results, MBIP complexity can be summarized in Table 2. We were able to prove complexity results for all the considered settings, but for  $w < n$  and

**Table 2** Computational complexity of MBIP

	$h \geq n$	$h < n$ fixed		$h < n$
$w < n$ fixed	<b>P</b> (Corollary 3)	<b>P</b> (Corollary 2)	$\Leftarrow$	<b>P</b> (Theorem 1)
$w < n$	?	<b>NP-hard</b> (Lemma 1)	$\Rightarrow$	<b>NP-hard</b> (Corollary 1)

$h \geq n$ , whose complexity is still unknown. Comparing Tables 1 and 2, it is easy to see that, for all the cases where the complexity of MBIP is known than it matches the corresponding BCPPG complexity.

### 4 Conclusions

We defined a new problem, the Minimum Blocking Item Problem, as a generalization of the Bounded Coloring Problem on Permutation Graphs. We studied the complexity of the proposed problem, providing results showing that it is NP-hard for  $w < n$  and  $h < n$ , fixed or not; moreover, we proved that it can be solved in polynomial time in all the other cases, but for  $w < n$  and  $h \geq n$ , whose complexity is still open.

**Acknowledgements** This work has been partially supported by Ministry of Instruction University and Research (MIUR) with the program PRIN 2015, project “SPORT—Smart PORT Terminals”, code 2015XAPRKF, and project “Nonlinear and Combinatorial Aspects of Complex Networks”, code 2015B5F27W.

### References

1. Bonomo, F., Mattia, S., Oriolo, G.: Bounded coloring of co-comparability graphs and the pickup and delivery tour combination problem. *Theoret. Comp. Sci.* **412**(45), 6261–6268 (2011)
2. Carlo, H.J., Vis, I.F.A., Roodbergen, K.J.: Storage yard operations in container terminals: literature overview trends and research directions. *Eur. J. Oper. Res.* **235**(2), 412–430 (2014)
3. Caserta, M., Schwarze, S., Voß, S.: A mathematical formulation and complexity considerations for the blocks relocation problem. *Eur. J. Oper. Res.* **219**, 96–104 (2012)
4. Izquierdo, C.E., Batista, B.M., Vega, J.M.M.: A domain-specific knowledge-based heuristic for the blocks relocation problem. *Adv. Eng. Inform.* **28**(28), 327–343 (2014)
5. Jansen, K.: The mutual exclusion scheduling problem for permutation and comparability graphs. *Inform. Comp.* **180**, 71–81 (2003)
6. Jovanovic, R., Voß, S.: A chain heuristic for the blocks relocation problem. *Comp. Ind. Eng.* **75**, 79–86 (2014)
7. Lehnfeld, J., Knust, S.: Loading unloading and premarshalling of stacks in storage areas: survey and classification. *Eur. J. Oper. Res.* **239**, 297–312 (2014)
8. Pnueli, A., Lempel, A., Even, W.: Transitive orientation of graphs and identification of permutation graphs. *Can. J. Math.* **23**, 160–175 (1971)

9. Tanaka, S., Takii, K.: A faster branch-and-bound algorithm for the block relocation problem. *IEEE Trans. Autom. Sci. Eng.* 7–12 (2014)
10. Wu, K.C., Ting, C.J.: A beam search algorithm for minimizing reshuffle operations at container yards. In: *International Conference on Logistics and Maritime Systems*, Busan, Korea, September 2010, pp. 15–17 (2010)
11. Zhu, W., Qin, H., Lim, A., Zhang, H.: Iterative deepening A algorithms for the container relocation problem. *IEEE Trans. Autom. Sci. Eng.* **9**, 710–722 (2012)

# A MILP Algorithm for the Minimization of Train Delay and Energy Consumption

Teresa Montrone, Paola Pellegrini and Paolo Nobili

**Abstract** A new timetable must be calculated in real-time when train operations are perturbed. The energy consumption is becoming a central issue both from the environmental and economic perspective but it is usually neglected in the timetable recalculation. In this paper, we formalize the real-time Energy Consumption Minimization Problem (rtECMP). The rtECMP is the real-time optimization problem of finding the driving regime combination for each train that minimizes the energy consumption, respecting given routing and precedences between trains. We model the trade-off between minimizing the energy consumption and the total delay by considering as objective their weighted sum. We propose an algorithm to solve the rtECMP, based on the solution of a mixed-integer linear programming (MILP) model. We test this algorithm on the Pierrefitte-Gonesse control area, which is a critical area in France with dense mixed traffic. In particular, we consider a one-hour traffic perturbation. In this situation, we take into account different routing and precedence possibilities and we solve the corresponding rtECMP. This experimental analysis shows the influence on the solution of the weights associated with energy consumption and delay in the objective function. The results show that the problem is too difficult to be solved to optimality in real time, but is indeed tractable.

---

T. Montrone (✉) · P. Nobili  
University of Salento, Via per Arnesano, 73100 Lecce, Italy  
e-mail: montrone@esteco.com

P. Nobili  
e-mail: paolo.nobili@unisalento.it

T. Montrone  
ESTECO S.p.A., AREA Science Park, Padriciano 99, 34149 Trieste, Italy

P. Pellegrini  
University Lille Nord de France, 59000 Lille, France  
e-mail: paola.pellegrini@ifsttar.fr

P. Pellegrini  
IFSTTAR, COSYS, LEOST, Rue Élisée Reclus, BP-70317, 59650,  
Villeneuve D'Ascq, 59666 Lille, France



**Keywords** Railway traffic management • Energy consumption  
Mixed-Integer linear programming

## 1 Introduction

In the railway system, the *timetable* is designed so that traffic can be smoothly operated. However, when unexpected events occur during operations, causing train delays, a new timetable must be computed in real-time. In practice, this computation is usually not fully automated: a dispatcher manually establishes routes and schedules to perform traffic operations minimizing delays.

The real-time Railway Traffic Management Problem (rtRTMP) is the problem of automatically establishing the train routing and scheduling in real-time, minimizing a function of the delay propagation. In the literature, several algorithms have been proposed for solving the rtRTMP [2, 3, 6, 9]. Typically, the rtRTMP does not consider energy consumption, which is becoming a central issue both from the environmental and economic perspectives. Hence, some authors propose approaches to integrate the minimization of energy consumption in real-time railway traffic operations. In particular, [11] presents a method to find the optimal speed profiles and control regimes for two trains running on a corridor. [1] proposes a fuzzy predictive control approach to find energy-efficient locomotive operations in real-time when speed limits change. [10] introduces a dual-speed curve approach for energy-saving operation of a high-speed train in absence of traffic.

In this paper, we define the real-time Energy Consumption Minimization Problem (rtECMP). Indeed, the energy consumption depends on the driving regimes followed by the trains. The rtECMP finds the driving regime combinations which minimize the energy consumption in real-time. These combinations consider as an input the routing and the precedences between the trains of an rtRTMP solution and comply with them. The objective of the problem is the minimization of the weighted sum of energy consumption and total delay.

Moreover, we propose the Train Driving Regime Combination-MILP algorithm (TDRC-MILP): a MILP-based algorithm for the rtECMP. It first calculates the trains travel times and the energy consumption corresponding to different combinations of driving regimes. Then, it solves the rtECMP through a MILP solver and returns the optimal solution. We propose an experimental analysis on instances representing traffic in the Pierrefitte-Gonesse control area in France. This control area corresponds to a complex junction with dense mixed traffic.

The rest of the paper is organized as follows. Section 2 describes the problem. Section 3 presents the algorithm and the MILP model. Section 4 reports the experimental analysis and Sect. 5 concludes the paper.

## 2 Problem Description

During train operations, unexpected events cause delays that often propagate due to the emergence of conflicts. A conflict occurs when two trains, running at their planned speed, would require incompatible block sections concurrently. Block sections are track portions delimited by signals and the block sections that share a track section are called *incompatible*. To ensure that the safety distance is always maintained between consecutive trains, when a train driver encounters a signal with a restrictive (different from green) aspect, he has  $n - 1$  block sections to stop the train where  $n$  is the number of aspects characterizing the signaling system [8]. According to the characteristics of the train, an immediate brake may or may not be necessary.

When conflicts emerge and trains need to brake or stop, a new timetable needs to be defined and followed, with new passing and stopping times and possibly with different train *routes*. Train routes are sequences of block sections that can be traversed between the trains' origins and destinations in the control area. The decision of the new train routes and precedences is made in the rtRTMP. Precedences are described through train *schedules*, which are the times at which the trains enter each block section of their routes. Typically, the rtRTMP minimizes delays considering the trains traveling as fast as possible through the control area dealt with. Indeed, there is a trade-off between delays and energy consumption.

The rtECMP is the real-time optimization problem of finding the new schedules and driving regime combination for each train that minimize delays and energy consumption, while respecting traffic management decisions. In the rtECMP, a solution of the rtRTMP is considered as an input and the trains driving regime combinations are optimized while satisfying it.

## 3 TDRC-MILP

The rtECMP problem is solved by means of TDRC-MILP, a first version of which was introduced in [7]. The stations in the control area are represented by block sections where the trains can stop. Moreover, we consider a three-aspect signaling system. For ease of formulation, we suppose that the visibility distance of all signals is 0, that is, a train entering a block section with a yellow aspect will have to stop at the end of it. Once stopped, the driver sees the signal opening the next block section and may immediately accelerate if the signal already turned green or yellow. The trains start at the maximum allowed speed if they enter the control area from a neighboring one, or at speed 0 if the origin of their route is a station within the control area. Each train can change the driving regime in predefined positions within the block sections belonging to its route.

The algorithm starts with the calculation of the travel times of trains and the energy consumption for each train and for each of its block sections. The trains travel times consist of the *running* and *clearing times*, which are, respectively, the times

needed by a train to travel on and to clear each block section. In particular, a block section is clear if the train's tail has exited it. Then, based on these values, the TDRC-MILP builds the MILP model and solves it.

In the definition of the algorithm we use the following notation:

$T$ : Set of trains that travel in the control area

$T'$ : Subset of trains that start from a station in the control area

$i, j, k, l$ : Respectively, a train, a block section, a train initial speed and a train driving regime combination

$B_i$ : Sequence of block sections along the route of train  $i$ ,  $\forall i \in T$

$S_i$ : Set of stations where train  $i$  must stop,  $\forall i \in T$

$CO$ : Set of allowed driving regime combinations

$s$ : Number of subsections in a block section

$V_{i0}$ : Initial speed for train  $i$  in the first block section of its route in the control area

$V_{ij}^0$ : Set of possible initial speeds for train  $i$  in block section  $j \in B_i$

$r_{ijkl}$ : Running time of train  $i$  on block section  $j$  if entering at speed  $k$  and using driving regime combination  $l$

$c_{ijkl}$ : Clearing time of train  $i$  on block section  $j$  if entering at speed  $k$  and using driving regime combination  $l$

$E_{ijkl}$ : Energy consumption of train  $i$  on block section  $j$  if entering at speed  $k$  and using driving regime combination  $l$

$r_j$ : Release time for block section  $j \in B$

$f_j$ : Route formation time for block section  $j \in B$

$j_i^\circ$ : First block section in  $B_i$ , that is, the origin of the route of train  $i$ ,  $\forall i \in T$

$j_i^*$ : Last block section in  $B_i$ , that is, the destination of the route of train  $i$ ,  $\forall i \in T$

$Q_{(ij)k}$ : Set containing pairs  $(k', l) \in (V_{ij}^0, CO)$  of train initial speeds and driving regime combinations that imply  $k$  as train final speed, for train  $i \in T$  and block section  $j \in B_i$

$P_{jj'}$ : Set containing the ordered pairs  $(i, i')$ ,  $i, i' \in T$ , such that  $i'$  traverses  $j' \in B_{i'}$  before  $i$  traverses  $j \in B_i$ , with  $j$  and  $j'$  incompatible block sections

$M$ : Big Constant  $\gg 0$

$w_{ij}$ : Originally scheduled time at which train  $i$  should enter  $j \in S_i$ ,  $\forall i \in T$ , that is, planned arrival time at the station

$e_i$ : Originally scheduled time at which train  $i$  should exit the last block section of its route,  $\forall i \in T$ , that is, planned arrival time at destination

$\alpha, \beta$ : Weights associated with the energy consumption and total delay in the objective function.

### 3.1 Pre-computation

The train travel times and the energy consumption depend on the infrastructure, the train initial speed and the driving regime combination. The algorithm calculates these values for all trains in each block section of their route, considering all the feasible driving regime combinations.

We suppose that the possible driving regimes are acceleration until the maximum possible speed, cruising at constant speed, coasting and deceleration. When accelerating, the maximum power is given to the engine to reach the maximum possible speed. When cruising, the speed is maintained constant and the acceleration is null (if the slope is null as well). When coasting, the engine is stopped and the train moves by inertia. Finally, when decelerating, the train brakes. The energy evolves differently during the four regimes depending on the tractive effort employed. In particular, during the last two regimes, the energy consumption is null. We refer the interested reader to [4] for more details.

We assume that each train can change its driving regime in some predefined points of each block section. The algorithm splits each block section into  $s$  subsections. For each of these subsections exactly one driving regime is chosen. For each block section  $j \in B_i$ , starting from the possible initial speeds  $k$ , the algorithm calculates running time  $r_{ijkl}$ , clearing time  $c_{ijkl}$  and energy consumption  $E_{ijkl}$  for each possible driving regime combination  $l$  and for each train  $i \in T$ .

### 3.2 Model Formulation

In the MILP model, the continuous variables are used for times:  $t_{ij}$  is the time at which train  $i$  enters block section  $j$ ;  $q_{ij}$  handles the possible stop duration of  $i$  at the red signal at the end of  $j$ ;  $d_i$  indicates the total delay of  $i$  at its intermediate stops and at its exit from the control area. The binary variables are the following:

$$y_{ijkl} = \begin{cases} 1 & \text{if } i \text{ enters } j \text{ with speed } k \text{ and traverses } j \text{ using combination } l \\ 0 & \text{otherwise,} \end{cases}$$

$$z_{ij} = \begin{cases} 1 & \text{if } i \text{ enters } j \text{ with a yellow signal} \\ 0 & \text{otherwise.} \end{cases}$$

The objective is the minimization of a weighted sum of energy consumption and total delay:

$$\min \alpha \left( \sum_{i \in T} \sum_{j \in B_i} \sum_{k \in V_{ij}^0} \sum_{l \in CO} E_{ijkl} y_{ijkl} \right) + \beta \sum_{i \in T} d_i. \tag{1}$$

In addition to the standard binary and non-negativity ones, the model includes several constraints which are described in the following.

$$\sum_{k \in V_{ij}^0} \sum_{l \in CO} y_{ijkl} = 1 \quad \forall i \in T, j \in B_i \tag{2}$$

$$\sum_{(k',l) \in Q_{(i)k}} y_{ijk'l} = \sum_{l \in CO} y_{ij'kl} \quad \forall i \in T, j \in B_i \setminus j_i^*, j' \text{ follows } j, \forall k \in V_{ij'}^0 \quad (3)$$

$$t_{ij'} - t_{ij} = \sum_{k \in V_{ij'}^0} \sum_{l \in CO} r_{ijkl} y_{ijkl} + q_{ij} \quad \forall i \in T, j \in B_i \setminus j_i^*, j' \text{ follows } j, j \notin S_i \quad (4)$$

$$t_{ij'} - t_{ij} = \sum_{k \in V_{ij'}^0} \sum_{l \in CO} r_{ijkl} y_{ijkl} + d_{ij} + q_{ij} \quad \forall i \in T, j \in S_i, j' \in B_i \setminus S_i, j' \text{ follows } j \quad (5)$$

$$\sum_{(k,l) \in Q_{(i)0}} y_{ijkl} = 1 \quad \forall i \in T, j \in S_i \quad (6)$$

$$t_{ij'} \geq w_{ij} \quad \forall i \in T, j \in S_i, j' \in B_i, j' \text{ follows } j \quad (7)$$

$$t_{ij} - t_{i'j'} \geq \sum_{k \in V_{ij'}^0} \sum_{l \in CO} (r_{i'j'kl} + c_{i'j'kl} + f_{j'} + r_{j'}) y_{i'j'kl} \\ \forall (i', i) \in P_{j'j}, j \in B_i, j' \in B_{i'}, j' \text{ incompatible with } j \quad (8)$$

$$Mz_{ij'} + t_{ij'} \geq t_{i'j'} + \sum_{k \in V_{ij'}^0} \sum_{l \in CO} (r_{i'j'kl} + c_{i'j'kl} + f_{j'} + r_{j'}) y_{i'j'kl} \\ \forall (i', i) \in P_{j'j'}, j'' \in B_i, j \text{ follows } j'', j' \in B_{i'}, j' \text{ incompatible with } j \quad (9)$$

$$\sum_{k \in Q_{(i)0}} y_{ij''kl} \geq z_{ij''} \quad \forall i \in T, j'' \in B_i \quad (10)$$

$$q_{ij} \leq Mz_{ij} \quad \forall i \in T, j \in B_i \quad (11)$$

$$\sum_{j \in S_i} (t_{ij} - w_{ij}) + (t_{ij_i^*} + \sum_{k \in V_{ij}^0} \sum_{l \in CO} y_{ij_i^*kl} r_{ij_i^*kl}) - e_i \leq d_i \quad \forall i \in T \quad (12)$$

Constraints (2) guarantee that each train follows exactly one combination of driving regimes in each block section. Constraints (3) and (4) ensure that the final speed in each block section is equal to the initial speed in the following one and that each train arrives at each block section after spending the necessary running time in the previous one, respectively. Constraints (5) and (6) state that each train stops at least for the minimum dwell time in each station where it has a scheduled stop. For Constraints (7), the train must not leave the station before its planned departure time. Thanks to Constraints (8), only one train at a time can traverse each block section. Recall that precedences between trains are inputs supplied by a rtRTMP solver. When

train  $i$  must traverse block section  $j$  and  $i'$  must traverse  $j'$ , if  $(i', i) \in P_{jj'}$  then  $i'$  must release  $j'$  before  $i$  can enter  $j$ . Constraints (9) and (10) impose that a train stops with red signal at the end of a block section when it has to give precedence to one or more other trains. For these constraints to be meaningful,  $M$  must be at least equal to the time distance between the time at which train  $i'$  exits from  $j'$  and the time at which train  $i$  enters  $j''$  for all  $i, i' \in T, j' \in B_{j'}, j'' \in B_i$ . Constraints (11) state that  $q_{ij}$  is 0 when a train does not have to stop at a red signal. Finally, Constraints (12) assign variables  $d_i$  according to the scheduled times of train  $i$  at the stations where it has a scheduled stop and at its destination.

## 4 Experimental Analysis

We implement TDRC-MILP in Java and integrate it with IBM ILOG CPLEX Optimization Studio V12.6.1 [5], with time and tree memory limit of one CPU hour and  $e^{75}/8$ , respectively. We run the experiments on an Intel(R) Core(TM) i7-4790 CPU @3.60 GHz, 8 cores, 16GB RAM with Ubuntu 16.04LTS Linux operating system.

We test the algorithm on the French Pierrefitte-Gonesse control area. This control area includes 89 track-circuits grouped into 79 block sections and 39 routes. The traffic is dense and during a weekday 336 trains travel on the infrastructure. The trains are of three types: high-speed trains, conventional passengers trains and freight trains. Each train has particular rolling stock characteristics, for example, their maximum speed are, respectively, 300 km/h, 160 km/h and 100 km/h, while their mass are 180 T, 1485 T and 180 T. We create a traffic perturbation by imposing that 20% of trains, randomly selected, enter the control area with a random delay between 5 and 15 min. We consider the 1-h scenario between 6.00 AM and 7.00 AM that includes 16 trains. For this scenario, the perturbation includes a total entrance delay of about 63 min (3806 s) suffered by 6 trains. We obtain 15 rtECMP instances by finding alternative feasible routes and precedence constraints through the rtRTMP solver RECIFE-MILP [9].

We split each block section included in the train routes in 3 subsections of equal length and vary the weights associated with energy consumption and delay in the objective function (1). First, we consider  $\alpha = 0, \beta = 1$ : the trains travel as fast as possible and no attention is paid to the energy consumption. Second, we set  $\alpha = 1, \beta = 0$  and, third, both values equal to 0.5.

The average results are reported in Table 1. The trends of energy consumption and delay follow the expectations: the higher the weight of a quantity in the objective function, the better the results in terms of this quantity and the worse the results in terms of the other. Three observations can be made based on these results. First, setting  $\beta = 0$  is probably not a wise choice, since in this case all variables  $d_i$  can be set as large as possible without any impact on the solution. From here, the average value of 1382400 which is equal to the variables upper bound (86400) multiplied by the number of trains in an instance (16). In future studies, we will replace this weight with a small but positive value to obtain more meaningful values of  $d_i$ 's. Second, with

**Table 1** Average results on 15 instances

Weights in (1.1)	Energy consumption (MJ)	Delay (s)	CPU time (s)	Gap %
$\alpha = 0, \beta = 1$	5824	10420	2183	0
$\alpha = 0.5, \beta = 0.5$	100	35823	1541	0
$\alpha = 1, \beta = 0$	100	1382400	1230	0

$\alpha \geq 0.5$  the energy consumed in the optimal solutions does not change. This means that  $\alpha = 0.5$  is large enough to give prominence to energy consumption minimization. Most likely this is due to the fact that energy consumption and delay assume a value of different orders of magnitude, and hence the impact of the weights is not balanced. In future works, we will normalize the two components of the objective function to overcome this issue. Third, the difficulty of the instances, if we look at the CPU time needed to solve them, increases with  $\beta$ . We don't have an explanation for this result yet, and we will investigate it deeply in future research. In particular, the first step we will make is the deep investigation of the impact of the weights of delay and energy consumption in the objective function.

## 5 Conclusion

In this paper, we have formalized the rtECMP as the problem of minimizing train delays and energy consumption given fixed routes and precedences. In a traffic management system, when trains operations are perturbed, the rtECMP receives these routes and precedences by an rtRTMP solver. Then, it computes the efficient driving regime combinations to be transmitted to the drivers.

After formalizing the problem, we presented TDRC-MILP. It is a MILP-based algorithm for solving the rtECMP to optimality. We tested TDRC-MILP on instances representing the peak-hour traffic in the French Pierrefitte-Gonesse control area. The results show that realistic instances are tractable, but the optimal solutions can hardly be found in real-time. In addition, the results give several hints for future research.

## References

1. Bai, Y., Ho, T.K., Mao, B., Ding, Y., Chen, S.: Energy-efficient locomotive operation for chinese mainline railways by fuzzy predictive control. *IEEE Trans. Intell. Transp. Syst.* **15**, 938–948 (2014)
2. Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M.: A tabu search algorithm for rerouting trains during rail operations. *Transp. Res. Part B* **44**, 175–192 (2010)

3. D'Ariano, A., Corman, F., Pacciarelli, D., Pranzo, M.: Reordering and local rerouting strategies to manage train traffic in real-time. *Transp. Sci.* **42**(4), 405–419 (2008)
4. Howlett, P.G., Pudney, P.J.: *Energy-Efficient Train Control*. Springer (1995)
5. IBM ILOG CPLEX Optimization Studio V12.6.1, 2014. [http://www.ibm.com/supportknowledgecenter/SSSA5P\\_12.6.1](http://www.ibm.com/supportknowledgecenter/SSSA5P_12.6.1). Accessed Nov 2016
6. Mannino, C., Lamorgese, L.: An exact decomposition approach for the real-time train dispatching problem. *Oper. Res.* **63**, 48–64 (2015)
7. Montrone, T., Pellegrini, P., Nobili, P.: Energy consumption minimization problem in a railway network. EWGT2016. Istanbul, Turkey (2016)
8. Pachl, J.: *Timetable Design Principles*, Chapter 2 in *Railway Timetable & Traffic*. Eurail Press (2008)
9. Pellegrini, P., Marlière, G., Pesenti, R., Rodriguez, J.: RECIFE-MILP: an effective MILP-based heuristic for the real-time railway traffic management problem. *IEEE Trans. Intell. Transp. Syst.* **16**, 2609–2619 (2015)
10. Song, Y., Song, W.: A novel dual speed-curve optimization based approach for energy-saving operation of high-speed trains. *IEEE Trans. Intell. Transp. Syst.* **17**, 1564–1575 (2016)
11. Wang, P., Goverde, R.M.P.: Multiple-phase train trajectory optimization with signalling and operational constraints. *Transp. Res. Part C* **69**, 255–275 (2016)



# A MILP Reformulation for Train Routing and Scheduling in Case of Perturbation

Paola Pellegrini, Grégory Marlière, Raffaele Pesenti  
and Joaquin Rodriguez

**Abstract** In this paper we propose a reformulation of RECIFE-MILP aimed at boosting the algorithm performance. RECIFE-MILP is a mixed integer linear programming based heuristic for the real-time railway traffic management problem, that is the problem of re-routing and rescheduling trains in case of perturbation in order to minimize the delay propagation. The reformulation which we propose exploits the topology of the railway infrastructure. Specifically, it capitalizes on the implicit relations between routing and scheduling decisions to reduce the number of binary variables of the formulation. In an experimental analysis based on realistic instances representing traffic in the French Pierrefitte-Gonesse junction, we show the performance improvement achievable through the reformulation.

**Keywords** Real-time railway traffic management problem  
Train re-routing and rescheduling · Mixed-integer linear programming

---

P. Pellegrini (✉) · G. Marlière · J. Rodriguez  
Université Lille Nord de France, 59000 Lille, France  
e-mail: paola.pellegrini@ifsttar.fr

P. Pellegrini  
LEOST, COSYS, IFSTTAR, 59650 Villeneuve d'Aascq, France

G. Marlière · J. Rodriguez  
ESTAS, COSYS, IFSTTAR, 59650 Villeneuve d'Aascq, France  
e-mail: gregory.marliere@ifsttar.fr

J. Rodriguez  
e-mail: joaquin.rodriguez@ifsttar.fr

R. Pesenti  
Department of Management, Università Ca' Foscari Venezia, 30121 Venezia, Italy  
e-mail: pesenti@unive.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_50

# 1 Introduction

Railway timetables extensively exploit the infrastructure for accommodating traffic, especially at peak hours and at critical locations. This extensive exploitation often translates into many trains traveling through critical junctions within short time horizons, where junctions are physical areas in which multiple lines cross. Indeed, unexpected events, even of apparently negligible entity, may cause a relevant deviation with respect to the scheduled timetable. In fact, according to the timetable, trains may be scheduled to traverse the same track segment at a very short time distance. If one of them is delayed due to an unexpected event, conflicts may emerge: multiple trains traveling at the planned speed would claim one or more track segments concurrently, and hence some of them have to stop or slow-down for ensuring safety. Conflicts may generate a severe delay propagation. In the practice, conflicts are tackled by dispatchers, who decide how to locally route and schedule trains based on their experience and on quite basic visual support tools. Due to the absence of advanced decision support tools, the decisions made by dispatchers may often be of a rather low quality if compared to what may be possible thanks to optimization.

In the literature, the selection of the train routes and schedules for minimizing delay propagation has been formalized as the real-time Railway Traffic Management Problem (rtRTMP) [8]. This is one of the main railway optimization problems which have been considered in the literature [5]. Several algorithms have been presented to tackle this problem [1–3]. Among them, we proposed RECIFE-MILP [7]. It is an algorithm based on the solution of a mixed-integer linear programming (MILP) formulation [8]. It has been validated on several case-studies coming from France [9, 11], Sweden, the UK and the Netherlands [10]. Despite the good performance achieved in all these case-studies, it is indeed possible to define instances for which RECIFE-MILP finds it difficult to return a high quality solution in real-time. The difficulty is very often linked to the size of the formulation describing the instances, which may easily include several tens of thousands of binary variables [7].

In this work, we propose an effective reformulation of the MILP formulation at the basis of RECIFE-MILP, which allows a strong reduction of the number of binary variables necessary to represent an instance. This can be done thanks to the exploitation of the links which exist between routing and scheduling variables, and through the reformulation of two sets of constraints. In the experimental analysis, we test this reformulation on perturbations of real instances representing traffic in the French control area of Pierrefitte-Gonesse. This analysis allows the quantification of the improvement of the performance of RECIFE-MILP in terms of decrease of the computational time necessary to prove the optimality of solutions and of improvement of solution quality (reduction of delay propagation) achieved when the optimality cannot be proven in the available computational time.

The rest of the paper is organized as follows. Section 2 describes the standard RECIFE-MILP formulation. Section 3 introduces the reformulation. Section 4 reports the computational experiments performed, and Sect. 5 our conclusions.

## 2 Standard RECIFE-MILP Formulation

In this section, we describe the main features of the MILP formulation that we use in RECIFE-MILP [7], in the next sections referred as to standard RECIFE-MILP. It models the infrastructure at the microscopic level and it implements the route-lock sectional-release interlocking system [6]. The objective function to be minimized is the total weighted delays suffered by trains at the intermediate stops and at their exit from the infrastructure. Although very often RECIFE-MILP quickly finds the optimal solution to realistic instances, it fails sometimes in delivering it within a computational time in line with real-time purposes. In this case, it stops the search process after the available time has elapsed and returns the best solution identified, together with the optimality gap.

In addition to ensure the coherent movement of each train along the infrastructure, RECIFE-MILP imposes disjunctive constraints to ensure the respect of the practical safety constraints. These are the constraints we consider in the reformulation, and hence the only ones we detail here for sake of brevity. We refer the interested reader to [7, 8] for all the details of the formulation and the algorithm.

The disjunctive constraints are imposed on each track-circuit, which are track sections on which the presence of a train is automatically detected. In a railway infrastructure, sequences of track-circuits are grouped into block sections, which are opened by a signal indicating their availability. Before a train can enter a block section, all the track-circuits belonging to the same block section must be reserved for the train itself. We name *utilization* time the sum of reservation and occupation time. The constraints are formulated as:

$$eU_{t,tc} - M(1 - y_{t,t',tc}) \leq sU_{t',tc} \forall t, t' \in T, \text{index } t < \text{index } t', tc \in TC_t \cap TC_{t'} \quad (1)$$

$$eU_{t',tc} - My_{t,t',tc} \leq sU_{t,tc} \forall t, t' \in T, \text{index } t < \text{index } t', tc \in TC_t \cap TC_{t'} \quad (2)$$

with:  $T$  set of trains;  $TC_t$  set of routes and track-circuits which can be used by train  $t$ ;  $M$  large constant;  $sU_{t,tc} \in \mathbb{R}^+$  time at which  $t$  starts utilizing  $tc$ ,  $\forall t \in T, tc \in TC_t$ ;  $eU_{t,tc} \in \mathbb{R}^+$  time at which  $t$  ends utilizing  $tc$ ,  $\forall t \in T, tc \in TC_t$ ;  $y_{t,t',tc} = 1$  if  $t$  utilizes  $tc$  before  $t'$ , 0 otherwise,  $\forall t, t' \in T : t < t', tc \in TC_t \cap TC_{t'}$ . These last variables need to be defined only for couples of trains  $t$  and  $t'$  such that the index of  $t$  is smaller than the one of  $t'$ , since  $y_{t',t,tc} = 1 - y_{t,t',tc}$ .

Remark that, if  $t$  chooses a route which includes  $tc$  and  $t'$  does not, then  $y_{t,t',tc} = 0$ :  $sU_{t,tc}$  and  $eU_{t,tc}$  are equal to the time at which  $t$  utilizes  $tc$ ;  $sU_{t',tc}$  and  $eU_{t',tc}$  are set to 0 by a set of constraints not detailed here, which ensure the coherence of the trains travel. Specifically, if a train does not use  $tc$ , its occupation and utilization (real) variables will be set to 0, as if the train traversed  $tc$ , and hence forbade other trains doing so, starting at time 0 and for 0 s. Conversely, if only  $t'$  chooses a route including  $tc$ , then  $y_{t,t',tc} = 1$ . If no train uses  $tc$  both values 0 and 1 are feasible for  $y_{t,t',tc}$ , since the utilization start and end will be 0 for both trains.

### 3 RECIFE-MILP Reformulation

In this section, we exploit the dependency of  $y_{t,t',tc}$  on the characteristics of the infrastructure and on the routes chosen by the trains to reformulate RECIFE-MILP with the aim of reducing the number of variables  $y_{t,t',tc}$ .<sup>1</sup> To this end, we first introduce the concepts of track-circuit equivalence and representativeness. Then, we reformulate Constraints (1) and (2) in terms of small sets of representative track-circuits. To do so, we will use  $R_t$  as the set of routes and track-circuits which can be used by train  $t$ , and  $TC^r$  as the set of track-circuits composing route  $r$ .

Given two trains  $t$  and  $t'$  and two routes  $r \in R_t$  and  $r' \in R_{t'}$ , we define the following relation in the set of the track-circuits  $TC^r \cap TC^{r'}$  shared by the two routes:

$$tc \sim_{r,r'} \hat{tc} \text{ if } tc, \hat{tc} \in TC^r \cap TC^{r'} \text{ and } t < t' \text{ on } tc \text{ iff } t < t' \text{ on } \hat{tc} \text{ for any feasible schedule.}$$

In other words, in any feasible solution of RECIFE-MILP such that  $x_{t,r} = x_{t',r'} = 1$ , with  $x_{t,r}$  binary variable stating whether train  $t$  chooses route  $r$  ( $= 1$ ) or not ( $= 0$ ), if  $tc \sim_{r,r'} \hat{tc}$  then  $y_{t,t',tc} = y_{t,t',\hat{tc}}$ : the precedence relation holding between  $t$  and  $t'$  on  $tc$  is necessarily the same as on  $\hat{tc}$ . As an example think of two consecutive track-circuits on a single track line. This relation is an equivalence one. Accordingly, it induces a partition of the set  $TC^r \cap TC^{r'}$  into equivalence classes. Here, an equivalence class  $S$  is a maximal subset of  $TC^r \cap TC^{r'}$  defined by the equivalence relation  $\sim_{r,r'}$ : it includes all track-circuits linked to each other by  $\sim_{r,r'}$ . Hereafter, we call *section* any of these equivalence classes  $S$ . We define the following sets for any pair of trains  $t$  and  $t'$  in  $T$ :

- $S^{r,r'} = \{S \subseteq TC^r \cap TC^{r'} : S \text{ equivalence class of } \sim_{r,r'}\}$  for all  $r \in R_t$  and  $r' \in R_{t'}$ , i.e.,  $S^{r,r'}$  is the set of sections associated to the pair of routes  $r$  and  $r'$ ;
- $S_{t,t',tc} = \{S \in S^{r,r'} : tc \in S, r \in R_t, r' \in R_{t'}\}$  for all  $tc \in TC_t \cap TC_{t'}$ , i.e.,  $S_{t,t',tc}$  is the set of all the sections which include  $tc$  given all the possible pairs of routes  $r$  and  $r'$  that  $t$  and  $t'$  may choose, respectively;
- $S_{t,t'} = \cup_{tc \in TC_t \cap TC_{t'}} S_{t,t',tc}$ , i.e.,  $S_{t,t'}$  is the set of all the sections for the two trains.

So far we have observed that, if  $tc \sim_{r,r'} \hat{tc}$ , variable  $y_{t,t',\hat{tc}}$  can play the role of variable  $y_{t,t',tc}$  provided that the two trains  $t$  and  $t'$  respectively choose routes  $r$  and  $r'$ . Next, we extend the concept of equivalence to cover all the possible route choices.

For any  $tc \in TC_t \cap TC_{t'}$ , we say that  $\hat{tc} \in TC_t \cap TC_{t'}$  is *representative* of  $tc$  if it satisfies the following conditions: (i) there exists at least a pair of routes  $r \in R_t$  and  $r' \in R_{t'}$  such that  $tc, \hat{tc} \in TC^r \cap TC^{r'}$ ; (ii) for all pairs of routes  $r \in R_t$  and  $r' \in R_{t'}$  such that  $tc, \hat{tc} \in TC^r \cap TC^{r'}$ , there exists  $S \in S^{r,r'}$  such that  $tc, \hat{tc} \in S$ , or equivalently  $tc \sim_{r,r'} \hat{tc}$ . Remark that any  $tc$  is always representative of itself. Moreover, the representativeness is a symmetric relation.

<sup>1</sup>For readability, in the following, with a slight abuse of notation, we say that we reformulate RECIFE-MILP instead of the formulation at the basis of the RECIFE-MILP algorithm.

As aimed, if  $\hat{tc}$  is representative of  $tc$  then  $y_{t,t',\hat{tc}}$  can play the role of  $y_{t,t',tc}$  for any pair of routes  $r \in R_t, r' \in R_{t'}$  such that  $tc, \hat{tc} \in TC^r \cap TC^{r'}$ . However, note that representativeness concerns only the pairs of routes that contain both track-circuits.

We call  $i$ -th representative set of  $tc$ ,  $Rep_{t,t'}^i(tc)$ , a subset of representative track-circuits  $\hat{tc}$  of  $tc$  that covers the set  $S_{t,t',tc}$ :  $Rep_{t,t'}^i(tc) \cap S \neq \emptyset$  for any  $S \in S_{t,t',tc}$ . Remark that at least the representative set  $\{tc\}$  exists for any  $tc \in TC_t \cap TC_{t'}$ . Let  $Rep_{t,t'}(tc)$  be the set of all possible representative sets of  $tc$ . Hence, for any pair of routes  $r \in R_t$  and  $r' \in R_{t'}$ , and for any  $Rep_{t,t'}^i(tc) \in Rep_{t,t'}(tc)$ , we can find a  $\hat{tc} \in Rep_{t,t'}^i(tc)$  such that  $y_{t,t',\hat{tc}}$  plays the role of  $y_{t,t',tc}$  if  $r$  and  $r'$  are the routes chosen.

We are now ready to determine the minimal set  $H_{t,t'}^*$  of representative track-circuits in  $TC_t \cap TC_{t'}$  such that there exists a  $Rep_{t,t'}^i(tc) \subseteq H_{t,t'}^*$  for all  $tc \in TC_t \cap TC_{t'}$ . To this end, for all  $tc \in TC_t \cap TC_{t'}$ ,  $S \in S_{t,t',tc}$ , let us define the following variables:

$$a_{tc,\hat{tc},S} = 1 \text{ if } \hat{tc} \text{ is a representative of } tc \text{ and } \hat{tc} \in S, \text{ 0 otherwise,}$$

$$z_{\hat{tc}} = 1 \text{ if } \hat{tc} \in H_{t,t'}^*, \text{ 0 otherwise}$$

with  $tc, \hat{tc} \in TC_t \cap TC_{t'}$  and  $S \in S_{t,t',tc}$ . The optimal solution of the following binary programming problem identifies the elements of  $H_{t,t'}^*$ :

$$\begin{aligned} \min \quad & \sum_{\hat{tc} \in TC_t \cap TC_{t'}} z_{\hat{tc}} \\ & \sum_{tc \in S} a_{tc,\hat{tc},S} z_{\hat{tc}} \geq 1, \quad \forall tc \in TC_t \cap TC_{t'}, S \in S_{t,t',tc} \\ & z_{\hat{tc}} \in \{0, 1\} \quad \forall \hat{tc} \in TC_t \cap TC_{t'} \end{aligned} \tag{3}$$

For each  $tc \in TC_t \cap TC_{t'}$ , we define the representative set for the reformulation of RECIFE-MILP,  $Rep_{t,t'}^*(tc)$ , as the minimum cardinality subset of  $H_{t,t'}^*$  such that for all  $S \in S_{t,t',tc}$ ,  $S \cap Rep_{t,t'}^*(tc) \neq \emptyset$ . In words, it is a minimal subset of  $H_{t,t'}^*$  that includes a representative track-circuit of  $tc$  for each section  $S$  that includes  $tc$  itself. The track-circuits  $tc \in H_{t,t'}^*$  are those for which a variable  $y_{t,t',tc}$  must be defined. The ones in  $Rep_{t,t'}^*(tc)$  are those which need to be associated to  $tc$  in the reformulation to capture the precedence relation between  $t$  and  $t'$  on  $tc$  itself along any pair of available routes.

Problem (3) is NP-hard as it reduces to the hitting set problem [4]. However, our experience suggests that its instances can be solved in a fraction of a second by any commercial solver even for rather large infrastructures. In addition, all the operations involving the definition of set  $H_{t,t'}^*$  can be performed off-line, once the alternative routes for the trains which may travel along the infrastructure are established *a priori*.

**Assumption 1** For a pair of trains  $t$  and  $t'$  which may both use track-circuit  $tc$ , let  $y_{t,t',tc} = 1$  if neither  $t$  nor  $t'$  chooses a route which includes  $tc$ .

With Assumption 1 we state that we choose to set  $y_{t,t',tc} = 1$  if no train uses  $tc$ , with no loss of optimality. In the explanation of Constraints (1) and (2), we discussed how, if only  $t'$  chooses a route which includes  $tc$ , then  $y_{t,t',tc}$  must be equal to 1. Conversely,

if no train uses  $tc$ , both values 0 and 1 are feasible. Hereafter, we will set  $y_{t,t',tc} = 1$  when  $t$  does not choose a route including  $tc$ , independently of what  $t'$  does.

**Theorem 1** *Constraints:*

$$eU_{t,tc} - M \left( 1 - y_{t,t',\hat{tc}} + \sum_{\substack{r \in R_t: \\ tc \in TC^r \wedge \hat{tc} \notin TC^r}} x_{t,r} + \sum_{\substack{r' \in R_{t'}: \\ tc \notin TC^{r'} \wedge \hat{tc} \in TC^{r'}}} x_{t',r'} \right) \leq sU_{t',tc} \quad (4a)$$

$$eU_{t',tc} - M \left( y_{t,t',\hat{tc}} + \sum_{\substack{r \in R_t: \\ tc \notin TC^r \wedge \hat{tc} \in TC^r}} x_{t,r} + \sum_{\substack{r' \in R_{t'}: \\ tc \in TC^{r'} \wedge \hat{tc} \notin TC^{r'}}} x_{t',r'} \right) \leq sU_{t,tc} \quad (4b)$$

$$\forall t, t' \in T, t < t', tc \in TC_t \cap TC_{t'}, \hat{tc} \in Rep_{t,t'}^*(tc),$$

can replace Constraints (1) and (2).

*Proof* Throughout this proof, let  $r$  and  $r'$  be the routes used by trains  $t$  and  $t'$ , respectively, i.e., the routes for which  $x_{t,r} = 1$  and  $x_{t',r'} = 1$ .

First, consider the case in which both  $r$  and  $r'$  include  $tc$  and  $\hat{tc}$ . In this case, Constraints (4a) and (4b) can replace (1) and (2) for the following reasons. By definition of representative track-circuit,  $y_{t,t',tc} = y_{t,t',\hat{tc}}$  since both  $tc$  and  $\hat{tc}$  are used. In addition, the sum of  $x$ -variables in (4a) and (4b) cancels out. Analogous argument holds for Constraints (4b).

Then, suppose that the route chosen by at least one train does not include both  $tc$  and  $\hat{tc}$ . In this case, Constraints (4a) and (4b) become negligible. We discuss only Constraints (4a), as symmetric arguments hold for Constraints (4b).

- If  $tc \notin TC^r$ , then (4a) is negligible as  $eU_{t,tc} = 0$  and the rest of the l.h.s. of the inequality is at most 0, whether  $t$  uses or not  $\hat{tc}$  and  $t'$  uses or not  $tc$  or  $\hat{tc}$ .
- If  $tc \in TC^r$ ,  $\hat{tc} \notin TC^r$ , then (4a) is negligible as  $\sum_{r \in R_t: tc \in TC^r \wedge \hat{tc} \notin TC^r} x_{t,r} = 1$  and the second part of the l.h.s. of the inequality is for sure negative, whether  $t'$  uses or not  $tc$  or  $\hat{tc}$ .
- If  $tc \in TC^r$ ,  $\hat{tc} \in TC^r$ ,  $\hat{tc} \notin TC^{r'}$ , then (4a) is negligible as  $y_{t,t',\hat{tc}} = 0$  as remarked right above this theorem.
- If  $tc \in TC^r$ ,  $\hat{tc} \in TC^r$ ,  $\hat{tc} \in TC^{r'}$ ,  $tc \notin TC^{r'}$ , then (4a) is negligible as  $\sum_{r' \in R_{t'}: tc \notin TC^{r'} \wedge \hat{tc} \in TC^{r'}} x_{t',r'} = 1$ .

So far we have shown that Constraints (4a) and (4b) can respectively replace (1) and (2) for a fixed  $\hat{tc}$  and a fixed pair of routes  $r \in R_t$  and  $r' \in R_{t'}$ . Indeed, the covering of  $tc$  by the elements of  $Rep_{t,t'}^*(tc)$  depends of the trains' route choices. Hence, we conclude the proof by observing that the above replacement can be made for any pair of routes  $r \in R_t$  and  $r' \in R_{t'}$ :

**Table 1** Summary of the results over 100 instances

KPI	Standard RECIFE-MILP	Boosted RECIFE-MILP
Mean computational time (s)	91	66
Optima proved	76	92
Best result achieved	82	98
Mean % gap (%)	9.87	2.61

- when  $tc \in TC^r \cap TC^{r'}$ , there exists  $S \in S^{r,r'} \subseteq S_{t,t',tc}$  such that if  $tc \in S$ , then the previous argument applies as, by definition,  $Rep_{t,t'}^*(tc)$  contains at least a representative track-circuit belonging to each section in  $S_{t,t',tc}$ ;
- when  $tc \notin TC^r \cap TC^{r'}$ , at least one train route does not include both  $tc$  and  $\hat{tc}$  for all  $\hat{tc} \in Rep^*(tc)$  and then, again, the previous argument applies.

□

In the light of the definition of  $H_{t,t'}^*$  and of Theorem 1, we can reformulate RECIFE-MILP by replacing Constraints (1) and (2) with Constraints (4a) and (4b) and by defining only the  $y$ -variables associated to track-circuits in  $H_{t,t'}^*$ , for each pair of trains  $t$  and  $t'$ . In the following, we will refer to the RECIFE-MILP algorithm using this reformulation as the boosted RECIFE-MILP.

## 4 Experimental Analysis

In this section, we report the results of an experimental analysis performed to compare the standard and the boosted RECIFE-MILP.

We consider instances representing traffic at the Pierrefitte-Gonesse junction, in France. It is a critical location with mixed traffic. The timetable of a week-day includes 340 trains crossing this control area: 120 high-speed and 129 conventional passenger trains, and 91 freight trains. Starting from this one-day timetable, we create 100 random scenarios: 20% of trains, randomly selected, suffer a random delay between 5 and 15 mins at their entrance in the control area. We generate one instance from each scenario by considering all the trains entering the control area between 7:00 and 8:00 am. This is the morning peak-hour, and each 1 h instance includes between 22 and 28 trains (26 on the average). Each train can use between 3 and 8 routes (6 on the average), which translates into a MILP formulation (for standard RECIFE-MILP) with about 17000 continuous variables, 3400 binary variables and 73000 constraints for an instance with 26 trains. No difference in the number of constraints or continuous variables exists between the standard and the boosted RECIFE-MILP. Instead, the number of  $y$ -variables goes from 2270 to 3754 (3119 on the average) for standard RECIFE-MILP, and from 800 to 1371 (1125 on the average) for the boosted one. We run the experiments on a computer with eight Intel

Xeon 3.5 Ghz processors and 128 GB RAM. The MILP solver used is IBM CPLEX MILP solver v 12.6, and the allowed computational time is three wall clock minutes, including the computation of the reformulation.

On the 100 instances solved, the boosted RECIFE-MILP obtains the best results under all the key performance indicators (KPIs) considered (Table 1).

## 5 Conclusions

In this paper we proposed a reformulation of RECIFE-MILP, a MILP-based heuristic algorithm for the real-time railway traffic management problem. This reformulation exploits the link between scheduling and routing decisions, for decreasing the number of binary variables of the MILP formulation.

We reported the results of an experimental analysis in which we compare the standard and the boosted RECIFE-MILP on realistic instances. We considered different KPIs, which all suggest that the reformulation proposed and implemented in the boosted RECIFE-MILP allows a clear performance improvement.

In future works, we will compare the performance of the two RECIFE-MILP versions on other case-studies to generalize this result. Moreover, we will propose sets of valid inequalities to further improve the performance.

## References

1. Cacchiani, V., Huisman, D., Kidd, M., Kroon, L., Toth, P., Veelenturf, L., Wagenaar, J.: An overview of recovery models and algorithms for real-time railway rescheduling. *Transp. Res. Part B Method.* **63**, 15–37 (2014)
2. Corman, F., Meng, L.: A review of online dynamic models and algorithms for railway traffic management. *IEEE Trans. Intel. Transp. Syst.* **16**(3), 1274–1284 (2015)
3. Fang, W., Yang, S., Yao, X.: A survey on problem models and solution approaches to rescheduling in railway networks. *IEEE Trans. Intel. Transp. Syst.* **16**(6), 2997–3016 (2017)
4. Karp, R.M.: Reducibility among combinatorial problems. In: *Complexity of Computer Computations*, pp. 85–103. Springer (1972)
5. Lusby, R., Larsen, J., Ehrgott, M., Ryan, D.: Railway track allocation: models and methods. *OR Spectr.* **33**(4), 843–883 (2011)
6. Pachel, J.: Timetable design principles. In: Hansen, I.A., Pachel, J. (eds.) *Railway Timetable and Traffic*, Chap. 2, pp. 9–42. Eurailpress | DVV Rail Media, Hambourg, Germany (2008)
7. Pellegrini, P., Marlière, G., Pesenti, R., Rodriguez, J.: RECIFE-MILP: an effective MILP-based heuristic for the real-time railway traffic management problem. *IEEE Trans. Intel. Transp. Syst.* **16**(5), 2609–2619 (2015)
8. Pellegrini, P., Marlière, G., Rodriguez, J.: Optimal train routing and scheduling for managing traffic perturbations in complex junctions. *Transp. Res. Part B Method.* **59**, 58–80 (2014)
9. Pellegrini, P., Marlière, G., Rodriguez, J.: A detailed analysis of the actual impact of real-time railway traffic management optimization. *J. Rail Transp. Plan. Manag.* **6**(1), 13–31 (2016)



10. Quaglietta, E., Pellegrini, P., Goverde, R.M.P., Albrecht, T., Jaekel, B., Marlière, G., Rodriguez, J., Dollevoet, T., Ambrogio, B., Carcasole, D., Giaroli, M., Nicholson, G.: The ON-TIME real-time railway traffic management framework: a proof-of-concept using a scalable standardised data communication architecture. *Transp. Res. Part C Em. Technol.* **63**, 23–50 (2016)
11. Samà, M., Pellegrini, P., D’Ariano, A., Rodriguez, J., Pacciarelli, D.: Ant colony optimization for the real-time train routing selection problem. *Transp. Res. Part B Method.* **85**, 89–108 (2016)

# **Part XI**

## **Routing**

# The Impact of a Clustering Approach on Solving the Multi-depot IRP

Luca Bertazzi, Annarita De Maio and Demetrio Laganà

**Abstract** We study the *Multi-Depot Inventory Routing Problem (MDIRP)* with homogeneous vehicle fleet and deterministic demand. We implement a branch-and-cut algorithm for this problem. Then, we design a matheuristic in which we first optimally solve a modified version of the *Capacitated Concentrator Location Problem (CCLP)* to generate a cluster of customers for each depot and, then, we exactly solve the problem based on these clusters with a branch-and-cut algorithm. Computational results are presented to compare the performance of the matheuristic with respect to the branch-and-cut, in order to analyze the value of the clustering approach in solving this problem.

**Keywords** Inventory routing · Branch-and-cut · Clustering

## 1 Introduction

*Inventory Routing Problems (IRPs)* spread out in the integrated optimization of inventory and distribution management in supply chains. This is a win-win approach in which a supplier coordinates the replenishment of a set of customers, deciding when to visit each customer over a time horizon, the quantities to deliver and the routes to travel. These problems received a remarkable attention in the recent decades. Examples of real industrial applications can be found in the survey pre-

---

L. Bertazzi

Department of Economics and Management, Università degli Studi di Brescia, Contrada S. Chiara n.50, 25122 Brescia, Italy  
e-mail: luca.bertazzi@unibs.it

A. De Maio (✉) · D. Laganà

Department of Mechanical, Energy and Management Engineering, Università della Calabria, Ponte Pietro Bucci 41c, 87036 Arcavacata di Rende (CS), Italy  
e-mail: annarita.demaio@unical.it

D. Laganà

e-mail: demetrio.lagana@unical.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217, DOI 10.1007/978-3-319-67308-0\_51

sented by [1]. Different versions of the problem were investigated in the literature, for example the single and multi-vehicle case, the single and multi-product case, the cases with deterministic, stochastic and robust demand. For an in depth analysis of the state of the art, the reader can refer to the tutorials by [5, 6] and to the survey by [9].

*IRPs* are well known to be NP-hard problems. For this reason, the main challenge is to design efficient exact algorithms on one side and effective heuristic algorithms on the other side. Different solution methods were proposed in the past for the single-depot case: exact methods can be found in [2, 10], while decomposition approaches are proposed by [8, 11]. The Multi-Depot *IRP* (*MDIRP*) is not largely investigated in literature. To the best of our knowledge, no branch-and-cut algorithms are available for it. A clustering technique to generate multi-depot instances was proposed by [15]. Instead, the multi-depot case was studied for the simpler *VRPs*, where the clustering of customers for each depot is often used. Effective clustering techniques are designed also for Location–Routing Problems. The most recent and complete survey about this class of problems is presented by [17] and an effective application can be found in [7].

Our contribution is to provide the first branch-and-cut algorithm for the *MDIRP* and to design a matheuristic algorithm, based on an optimal solution of a variant of the classical *Capacitated Concentrator Location Problem* (*CCLP*). Computational results are presented to evaluate the impact of the clustering on the solution quality and on the computational time, with respect to the best solution provided by the branch-and-cut.

The remainder of the paper is organized as follows. The *MDIRP* is formally described and formulated in Sect. 2. The branch-and-cut algorithm is described in Sect. 3. The matheuristic we propose is described in Sect. 4. The computational results are shown in Sect. 5. Finally, conclusions are drawn in Sect. 6.

## 2 Problem Description and Formulation

In this section, the *MDIRP* is described and the corresponding mathematical formulation based on binary edge-variables is presented. This formulation is an extension of the single–vehicle and single–depot *IRP* formulation proposed by [2]. We consider a complete undirected graph  $G(V, E)$ . A set of depots  $P = \{1, 2, \dots, l\}$  delivers a product to a set  $I = \{1, 2, \dots, n\}$  of customers. The set with all vertices is denoted by  $V = P \cup I$ . The parameter  $c_{ij}$  is the travelling cost of the edge  $(i, j) \in E$ . These costs satisfy the triangle inequality. Given  $S$  (a proper and non–empty subset of vertices),  $S \in I$ ,  $E(S)$  denotes the set of edges  $(i, j)$ , such that  $i, j \in I$ . The set of the vehicles is denoted by  $K = \{1, 2, \dots, M\}$ . Each vehicle has capacity  $C$ . A discrete time horizon  $H$  is given. The set of time periods is denoted by  $T = \{1, 2, \dots, H\}$ . Each customer  $i \in I$  defines a maximum inventory level  $U_i$  and has a given start-

ing inventory level  $Inv_{i0} \leq U_i$ . At each time  $t \in T$ , each customer  $i$  has to satisfy the deterministic demand  $d_{it}$ . The variable  $Inv_{it}$  indicates the inventory level of customer  $i$  at the begin of period  $t$ . The time  $H + 1$  is included in the inventory computation to take into account the consequences of the decisions at time  $H$ . The variable  $y_{iktp}$  represents the quantity delivered to customer  $i$  in period  $t$  by vehicle  $k$  from depot  $p$ . The binary variable  $x_{ijkt}$  is equal to 1 if the edge  $(i, j)$  is traversed in period  $t$  by vehicle  $k$  starting from depot  $p$ . The binary variable  $z_{iktp}$  is equal to 1 if customer  $i$  is visited in period  $t$  by vehicle  $k$  from depot  $p$ . The binary variable  $z_{pkt}$  is equal to 1 if vehicle  $k$  located in depot  $p$  starts its tour from depot  $p$  in period  $t$ . This problem can be formulated as follows:

$$Min \sum_{t \in T} \sum_{i \in V} \sum_{j \in V} \sum_{k \in K} \sum_{p \in P} c_{ij} x_{ijkt} \tag{1}$$

s.to.

$$Inv_{i,t+1} = Inv_{it} + \sum_{k \in K} \sum_{p \in P} y_{iktp} - d_{it} \quad t \in T, i \in I \tag{2}$$

$$\sum_{p \in P} \sum_{k \in K} y_{iktp} + Inv_{it} \leq U_i \quad t \in T, i \in I \tag{3}$$

$$\sum_{i \in I} y_{iktp} \leq C z_{pkt} \quad t \in T, p \in P, k \in K \tag{4}$$

$$y_{iktp} \leq C z_{iktp} \quad i \in I, t \in T, p \in P, k \in K \tag{5}$$

$$\sum_{i \in I} y_{iktp} \geq z_{pkt} \quad t \in T, p \in P, k \in K \tag{6}$$

$$z_{bkt} = 0 \quad \forall t \in T, p \in P, k \in K, b \in P(p \neq b) \tag{7}$$

$$\sum_{p \in P} z_{pkt} \leq 1 \quad t \in T, k \in K \tag{8}$$

$$\sum_{p \in P} z_{iktp} \leq 1 \quad t \in T, k \in K, i \in I \tag{9}$$

$$\sum_{j \in I, j < i} x_{ijkt} + \sum_{j \in I, j > i} x_{ijkt} = 2 z_{iktp} \quad i \in I, t \in T, p \in P, k \in K \tag{10}$$

$$\sum_{(i,j) \in E(S)} x_{ijkt_p} \leq \sum_{i \in S} z_{iktp} - z_{ukt_p} \quad S \subseteq I, |S| \geq 2, t \in T, \quad (11)$$

$$k \in K, p \in P, \text{ for a given } u \in S$$

$$x_{ijkt_p} \in \{0, 1\} \quad i, j \in V, t \in T, p \in P, k \in K \quad (12)$$

$$Inv_{it} \geq 0 \quad i \in I, t \in T \quad (13)$$

$$y_{ikt_p} \geq 0 \quad i \in I, t \in T, p \in P, k \in K \quad (14)$$

$$z_{ikt_p} \in \{0, 1\} \quad i \in I, t \in T, p \in P, k \in K. \quad (15)$$

The objective function (1) states the minimization of the total routing cost. Constraints (2)–(3) are inventory constraints at the customers. Constraints (4)–(5) are capacity constraints. Constraints (6)–(9) define the multi-depot case and split delivery. Constraints (10)–(11) are classical routing constraints. Constraints (12)–(15) define integrality and non-negativity variables conditions.

### 3 A Branch-and-Cut Algorithm

In order to exactly solve the *MDIRP* described in the previous section, we design and implement the following branch-and-cut algorithm. The *subtour elimination constraints* (11) were initially removed from the formulation (1)–(15) and added dynamically using the separation procedure described in [16]. They were introduced considering a given  $u \in S$ , for which the following condition is valid:  $u = \operatorname{argmax}_i \{\bar{z}_{ikt_p}\}$ , where  $\bar{z}_{ikt_p}$  is the value of variable  $z_{ikt_p}$  in the current LP relaxation. At each tree node, the violated (11) are found and added to the current sub-problem that is then optimized. If no violations are identified, branching occurs at the current node. No priority variables are defined for the branching strategy. In order to improve the quality of the root node lower bound of the branch-and cut tree, the following valid inequalities are added to the initial *LP*.

1. *Priority inequalities*:

$$z_{ikt_p} \leq z_{pkt_p} \quad i \in I, t \in T, p \in P, k \in K. \quad (16)$$

These valid inequalities are used by [2]. We consider an adapted version of them for the *MDIRP*.

2. *Logical inequalities:*

$$x_{ipktp} + x_{piktp} \leq 2 z_{ikt p} \quad i \in I, t \in T, p \in P, k \in K. \quad (17)$$

$$x_{ijkt p} \leq z_{ikt p} \quad i, j \in I, t \in T, p \in P, k \in K. \quad (18)$$

These inequalities are inspired by the logical cuts by [12, 13].

3. *Aggregate parity inequalities:*

$$\sum_{p \in P} \sum_{(i,j) \in \delta(S)} x_{ijkt p} \geq \sum_{p \in P} \sum_{(i,j) \in F} x_{ijkt p} - |F| + 1, \quad t \in T, k \in K, \quad (19)$$

$$F \subseteq \delta(S), |F| \text{ odd.}$$

4. *Disaggregate parity inequalities:*

$$\sum_{(i,j) \in \delta(S)} x_{ijkt p} \geq \sum_{(i,j) \in F} x_{ijkt p} - |F| + 1, \quad t \in T, p \in P, k \in K, \quad (20)$$

$$F \subseteq \delta(S), |F| \text{ odd.}$$

*Parity inequalities* are initially defined in [4] as co-circuit inequalities. They are really effective for problems with binary variables, in case the parity of vertices is required. Inequalities (19)–(20) are separated heuristically following the procedure described by [3].

## 4 A Matheuristic for the *MDIRP*

As explained before, solving *MDIRP* with an exact method is really complex. The branch-and-cut algorithm is very slow even in small instances. Therefore, we design a matheuristic algorithm based on the following three steps:

1. Optimally solve the *CCLP* described below to generate clusters composed by a depot and a subset of customers.
2. Import the clusters in the *MDIRP* model: for each customer  $i$  not associated with depot  $p$  the corresponding  $z_{ikt p}$  variables are set to zero, in order to forbid routes serving customers not associated to  $p$ .
3. Apply the branch-and-cut described in Sect. 3 to the model obtained in step 2.

Let us now describe the *CCLP* we solve in the matheuristic. The *CCLP* was largely investigated in the literature for the *VRP*. Different formulations and variants are described by [14]. We present an adapted formulation of the classical *CCLP* in order to match with the *MDIRP* case. We define the set  $P$  of depots as the set of potential concentrators. Let  $\gamma$  be the number of concentrators to open,  $R_p$  be the fixed cost to make  $p$  as a concentrator,  $D_i$  be the demand of customer  $i$  used to build the clusters,

$\Gamma_{pi}$  be the cost to assign customer  $i$  to concentrator  $p$ ,  $\Phi_p$  be capacity of each concentrator. The model involves two sets of binary variables:  $g_{pi}$  equal to 1 if the customer  $i$  is assigned to depot  $p$  and  $b_p$  equal to 1 if  $p$  is selected to be a concentrator. The mathematical formulation is described below:

$$\text{Min } \sum_{p \in P} \sum_{i \in I} \Gamma_{pi} g_{pi} + \sum_{p \in P} R_p b_p \quad (21)$$

s.t.

$$\sum_{p \in P} g_{pi} = 1 \quad i \in I \quad (22)$$

$$\sum_{i \in I} D_i g_{pi} \leq \Phi_p b_p \quad p \in P \quad (23)$$

$$\sum_{p \in P} b_p = \gamma \quad (24)$$

$$b_p \in \{0, 1\} \quad p \in P \quad (25)$$

$$g_{pi} \in \{0, 1\} \quad p \in P, \quad i \in I. \quad (26)$$

The objective function (21) minimizes the total cost to build the clusters. Constraints (22) guarantee that each customer is assigned to exactly one cluster. Constraints (23) ensure that the capacity of each cluster is not violated. Constraint (24) guarantee that  $\gamma$  clusters are built. Constraints (25)–(26) impose that the variables are all binary.

## 5 Computational Results

The branch-and-cut and the matheuristic described in Sects. 3 and 4 were implemented in C++ by using IBM Concert Technology and CPLEX 12.6, and run on an Intel Core i7-6500U 2.50 GHz and 8 GB RAM personal computer. An adapted version of two subsets of instances derived from the benchmark instances provided in [2] for the single-depot IRP are used. Instances are labeled as  $nNdDhH$ , where  $N$  is the number of customers,  $D$  is the number of the depots,  $H$  is the time horizon. A time limit of 2 hours is set for both the branch-and-cut and the matheuristic.



**Table 1** CCLP

Instance	Time (ms)	N. Clusters	Cardinality	Instance	Time (ms)	N. Clusters	Cardinality
n5d2h3	40	2	3-2	n5d2h3	30	2	3-2
n10d2h3	50	2	4-6	n10d2h3	40	2	4-6
n15d2h3	70	2	6-9	n15d2h3	35	2	8-7
n20d30h3	50	3	5-11-4	n2d30h3	70	3	4-12-4
n25d4h3	60	4	5-9-4-7	n25d4h3	50	4	9-5-4-7
n30d4h3	80	4	5-12-4-9	n30d4h3	70	4	5-11-5-9
n5d2h6	40	2	3-2	n5d2h6	30	2	3-2
n10d2h6	50	2	4-6	n10d2h6	55	2	3-7
n15d2h6	30	2	11-4	n15d2h6	30	2	11-4
n20d3h6	90	3	4-9-7	n20d3h6	80	3	12-4-4

Table 1 provides the results obtained by solving the *CCLP* with the following data:  $\gamma = |P|$ ,  $R_p = 0$ ,  $D_i$  equal to the average demand over the time horizon,  $\Gamma_{pi} = c_{pi}$ ,  $\Phi_p = MC$ . For each instance, it gives the computational time (ms), the number of clusters and their cardinality.

The results show that the clustering phase is no time consuming and the cardinality of the clusters is enough homogeneous in each instance.

Table 2 compares the results obtained by applying the branch-and-cut algorithm introduced in Sect. 3 and the matheuristic described in Sect. 4. For each instance, it shows, both for the matheuristic and the branch-and-cut, the computational time (seconds) or t.l. when the time limit is reached, the total number of added inequalities (subtour elimination constraints, disaggregated and aggregated parity inequalities), the cost of the best feasible solution found in the time limit, the lower bound value and the CPLEX GAP. In the last column the percentage GAP between the feasible solutions of two approaches is shown.

The results show that the matheuristic is always able to find a feasible solution in the time limit, while the branch-and-cut is not able to find it in 35% of the instances. Moreover, the CPLEX GAP of the matheuristic is about 9% on average, while it is about 26% in the solved instances. Note that the CPLEX GAP increases in the instances with time horizon  $H = 6$  with respect to the instances with  $H = 3$ . The last column of Table 2 shows that the matheuristic is able to find better solutions than the branch-and-cut in the same time limit, with a maximum reduction of about 23% of the cost. This underlines that the matheuristic is more efficient in terms of solution quality and computational time.

**Table 2** Matheuristic vs. Branch-and-cut

Instance	Matheuristic						Branch-and-cut						Gap (%)
	Time (s)	N.Soub.	N.Par.	Cost	LB	Gap	Time (s)	N.Soub.	N.Par.	Cost	LB	Gap	
n5d2h3	51.35	22	198	1148.8	1148.8	0.00	210.77	127	436	1148.8	1148.8	0.00	0.00
n10d2h3	1150.83	230	677	2214.29	2214.29	0.00	t.l.	1008	1295	2149.16	1972.55	8.22	3.00
n15d2h3	t.l.	554	925	4117.12	3783.77	8.10	t.l.	1785	1802	4614.66	3383.05	26.69	-11.00
n20d3h3	t.l.	572	992	3467.5	3205.57	7.55	t.l.	1604	1954	3629.04	2579	28.93	-4.00
n25d4h3	t.l.	369	728	3917.78	3536.76	9.04	t.l.	/	/	/	/	/	/
n30d4h3	t.l.	603	1185	4401.29	4142.78	5.87	t.l.	/	/	/	/	/	/
n5d2h6	5.06	51	185	2705.04	2705.04	0.00	5003.34	414	1035	2595.14	2595.14	0.00	4.00
n10d2h6	t.l.	444	1245	4939.37	4270.29	13.55	t.l.	1295	1610	5117.57	2971.66	41.93	-3.00
n15d2h6	t.l.	1264	2811	10830.5	7601.98	29.81	t.l.	1725	3565	10850	6795.24	37.37	-0.10
n20d3h6	t.l.	723	2278	11242.9	8899.94	20.84	t.l.	/	/	/	/	/	/
Instance	Time (s)	N.Soub.	N.Par.	Cost	LB	GAP	Time (s)	N.Soub.	N.Par.	Cost	LB	GAP	
n5d2h3	106.17	27	216	956.38	956.30	0.00	112.79	98	314	956.38	956.38	0.00	0.00
n10d2h3	t.l.	173	560	2572.43	2346.81	9.61	t.l.	718	1072	2653.39	1789.64	48.27	-3.05
n15d2h3	t.l.	325	649	2436.43	2202.38	10.62	t.l.	573	1080	2436.43	2154.64	13.07	0.00
n20d3h3	3048.27	336	837	3837.28	3837.28	0.00	t.l.	1022	2005	4969.69	3460.06	43.63	-22.78
n25d4h3	t.l.	360	472	3917.78	3536.76	9.04	t.l.	/	/	/	/	/	/
n30d4h3	t.l.	493	784	4415.79	4118.8	7.21	t.l.	/	/	/	/	/	/
n5d2h6	t.l.	44	704	5243.49	5218.33	0.48	t.l.	384	932	6304.28	4289.37	46.97	-16.82
n10d2h6	t.l.	458	991	5530.09	4926.71	12.24	t.l.	1141	1793	6055.99	3609.7	67.97	-8.68
n15d2h6	t.l.	864	1901	9047.82	6880.83	31.49	t.l.	/	/	/	/	/	/
n20d3h6	t.l.	726	1512	6748.37	5884.97	12.97	t.l.	/	/	/	/	/	/

## 6 Conclusion

We studied the *Multi-Depot Inventory Routing Problem (MDIRP)*. This problem is very difficult to be solved to optimality. A branch-and-cut algorithm designed for it was able to find a feasible solution in only 65% of the instances and provides an average CPLEX GAP 26% of in the solved instances. Our results showed that embedding the clusters obtained by optimally solving a variant of the *Capacitated Concentrator Location Problem* in the branch-and-cut allowed us to always find a feasible solution of the problem, to reduce the CPLEX GAP to 9% and to find better solutions. Future research could be devoted to improve this matheuristic, trying to strongly enhance the clustering approach.

## References

1. Andersson, H., Hoff, A., Christiansen, M., Hasle, G., Løkketangen, A.: Industrial aspects and literature survey: combined inventory management and routing. *Comput. Oper. Res.* **37**(9), 1515–1536 (2010)
2. Archetti, C., Bertazzi, L., Laporte, G., Speranza, M.G.: A branch-and-cut algorithm for a vendor-managed inventory-routing problem. *Transport. Sci.* **41**(3), 382–391 (2007)
3. Aráoz, J., Fernández, E., Meza, O.: Solving the prize-collecting rural postman problem. *Eur. J. Oper. Res.* **196**(3), 886–896 (2009)
4. Barahona, F., Grötschel, M.: On the cycle polytope of a binary matroid. *J. Comb. Theory Ser. B* **40**(1), 40–62 (1986)
5. Bertazzi, L., Speranza, M.G.: Inventory routing problems: an introduction. *EURO J. Transp. Log.* **1**, 307–326 (2012)
6. Bertazzi, L., Speranza, M.G.: Inventory routing with multiple customers. *EURO J. Transp. Log.* **2**, 255–275 (2013)
7. Bramel, J., Simchi-Levi, D.: A location based heuristic for general routing problems. *Oper. Res.* **43**(4), 649–660 (1995)
8. Campbell, A.M., Savelsbergh, M.: A decomposition approach for the inventory routing problem. *Transp. Sci.* **38**(4), 488–502 (2004)
9. Coelho, L.C., Cordeau, J.F., Laporte, G.: Thirty years of inventory-routing. *Transp. Sci.* **48**, 1–19 (2013)
10. Coelho, L.C., Laporte, G.: Improved solutions for inventory-routing problems through valid inequalities and input ordering. *Internat. J. Prod. Econ.* **155**, 391–397 (2014)
11. Cordeau, J.-F., Laganà, D., Musmanno, R., Vocaturo, F.: A decomposition-based heuristic for the multiple-product inventory-routing problem. *Comp. Oper. Res.* **55**, 153–166 (2015)
12. Fischetti, M., Gonzalez, J.J.S., Toth, P.: Solving the orienteering problem through branch-and-cut. *J. Comp.* **10**(2), 133–148 (1998)
13. Gendreau, M., Laporte, G., Semet, F.: A branch-and-cut algorithm for the undirected selective travelling salesman problem. *Networks* **32**(4), 263–273 (1998)
14. Gouveia, L., Saldanha-da-Gama, F.: On the capacitated concentrator location problem: a reformulation by discretization. *Comp. Oper. Res.* **33**, 1242–1258 (2006)
15. Noor, N. M. and Shuib, A.: Multi-depot instances for inventory routing problem using clustering techniques. *J. Ind. Intell. Inf.* **3**(2), 97–101 (2015)
16. Padberg, M., Rinaldi, G.: A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Rev.* **33**(1), 60–100 (1991)
17. Prodhon, C., Prins, C.: A survey of recent research on location-routing problems. *Eur. J. Oper. Res.* **238**, 1–17 (2014)

# Optimal Paths for Dual Propulsion Vehicles on Real Street Network Graphs

Giovanni Capobianco, Carmine Cerrone, Raffaele Cerulli  
and Giovanni Felici

**Abstract** There are several examples of dual propulsion vehicles: hybrid cars, bi-fuel vehicles, electric bikes. Compute a path from a starting point to a destination for these typologies of vehicles requires evaluation of many alternatives. In this paper we develop a mathematical model, able to compute paths for dual propulsion vehicles, that takes in account the power consumption of the two propulsors, the different types of charging, the exchange of energy and, last but not least, the total cost of the path. We focus our attention on electric bikes and we perform several experiments on real street network graph. In our tests we took into account the slope of roads, the recharge in downhill streets and the effort of the cyclist. To validate the model we performed computational tests on properly generated instances set. This set of instances is composed of graphs representing real cities of all around the world. The computational tests show the effectiveness of our approach and its applicability on a real street network.

## 1 Introduction

Hybrid cars, bi-fuel vehicles, electric bikes, are examples of dual propulsion vehicles. In some cases, one propulsor can recharge the other propulsion, or energy could be recovered (downhill roads, braking). Sachenbacher et al. [11], proposed

---

G. Capobianco · C. Cerrone (✉)  
University of Molise, contrada Fonte Lappone, 86090 Pesche (IS), Italy  
e-mail: carmine.cerrone@unimol.it

G. Capobianco  
e-mail: giovanni.capobianco@unimol.it

R. Cerulli  
University of Salerno, via Giovanni Paolo II 138, 84084 Fisciano, Italy  
e-mail: raffaele@unisa.it

G. Felici  
Istituto di Analisi dei Sistemi e Informatica, via dei Taurini 19, 00185 Rome, Italy  
e-mail: giovanni.felici@iasi.cnr.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_52

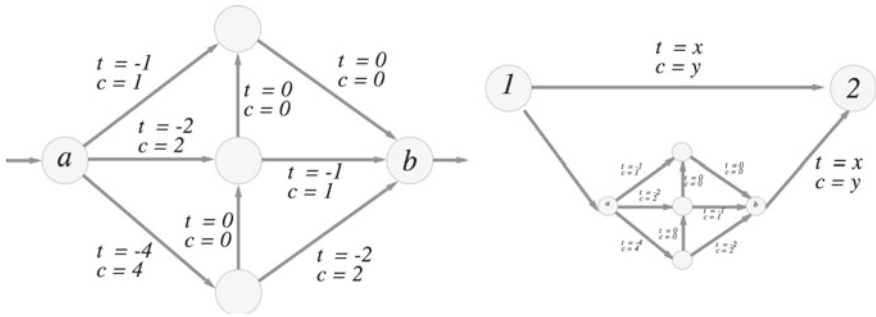
an A\* framework to optimize the routing for electric vehicles. Baum et al. [1, 2] proposed a routing algorithm that takes into accounts the energy recovery and the battery capacity by minimizing energy consumption in order to maximize cruising range. Zundorf [13] proposed a heuristic approach to design path for electric vehicles considering the position of the charging stations. Salimifard and Raeesi [12] proposed a variant of the vehicle routing problem, the green routing problem (GRP) which deals with optimizing CO<sub>2</sub> emissions and costs using a bi-fuel vehicle fleet.

In this paper, we propose a mathematical model based on a flow formulation, able to compute a path taking into account: the energy consumption of the two propulsors, the refueling or recharging, the exchange of energy, the recuperation of energy the battery capacity and the cost of the path. The main contribution of this paper is to show the applicability of an exact approach on real street network graphs taking into accounts the complexity of dual propulsion vehicles. The remainder of the paper is organized as follow. In Sect. 2, we introduce our MIP model, in Sect. 3 we show our computational results, in Sect. 4 we summarize the results and give some conclusions.

## 2 Mathematical Model

In this section, we focus on electric bikes. In this scenario we have to consider an electric motor and the human propulsion. Let  $G(V, A)$  be a graph in which  $V$  is the set of vertices and  $A$  is the set of arcs. For each arc  $(i, j) \in A$ ,  $c_{i,j}$  is the cost of traversing the arc  $(i, j)$ ,  $t_{i,j}$  is the amount of energy used for traversing the arc. If  $t_{i,j} < 0$  the vehicle is recharging when using arc  $(i, j)$ . With the set of variables  $t$  is possible to model the charge of the battery on the downhill roads and the charging stations with different charging time depending on the amount of energy charged (Fig. 1).

The set of variables  $u$  is associated with the effort of the cyclist to give propulsion to the bike. The set of variables  $\hat{u}$ , is associated with the effort of the cyclist to recharge the battery the parameter  $k$  is used to convert the kinetic energy into electric energy. The set of boolean variables  $y$ , defined for each arc  $(i, j) \in A$ , is used to indicate if an arc is used in the solution ( $y_{i,j} = 1$ ). Variables  $v$ , are associated to the amount of energy used by the electric motor, while variables  $f$  are the arc's flow variables. The flow  $f_{i,j}$  on the arc  $(i, j)$  indicates the residual charge of the battery after crossing the arc. The vertices  $s$  and  $p$  are the starting and destination points of the path. The constant  $SP$  is the length of the shortest path from  $s$  to  $p$ . The constant  $max$  represents the maximum effort feasible for the cyclist. The constant  $px$  is a percentage that indicates the maximum allowed difference between the value  $SP$  and the optimal solution of the model in terms of path length.  $M$  is the capacity of the electric battery,  $\hat{M} \leq M$  is level of charge of the battery at the starting vertex  $s$ .



**Fig. 1** The subgraph (on the left) composed of five vertices can be used to describe the charging stations. It is possible to recharge different amount of energy, paying a cost depending on the charge. On the right an example of charging station on the arc 1–2

We can now present our mathematical model.

$$\text{MIN} \sum_{(i,j) \in E} u_{i,j} + k \sum_{(i,j) \in E} \hat{u}_{i,j} \tag{1}$$

subject to

$$v_{i,j} + u_{i,j} + \hat{u}_{i,j} = t_{i,j} y_{i,j} \quad \forall (i,j) \in E \tag{2}$$

$$\sum_{(j,i) \in E} f_{j,i} - \sum_{(i,j) \in E} f_{i,j} \geq \sum_{(j,i) \in E} v_{j,i} \quad \forall i \in V \setminus \{s,p\} \tag{3}$$

$$\sum_{(s,j) \in E} f_{s,j} \leq \hat{M} \tag{4}$$

$$\sum_{(j,s) \in E} f_{j,s} = 0 \tag{5}$$

$$\sum_{(p,j) \in E} f_{p,j} = 0 \tag{6}$$

$$\sum_{(j,i) \in E} y_{j,i} \leq 1 \quad \forall i \in V \setminus \{s,p\} \tag{7}$$

$$\tag{8}$$

$$\sum_{(i,j) \in E} y_{i,j} \leq 1 \quad \forall i \in V \setminus \{s, p\} \quad (9)$$

$$\sum_{(i,j) \in E} y_{i,j} - \sum_{(i,j) \in E} y_{i,j} = 0 \quad \forall i \in V \setminus \{s, p\} \quad (10)$$

$$\sum_{(s,j) \in E} y_{s,j} = 1 \quad (11)$$

$$\sum_{(j,p) \in E} y_{j,p} = 1 \quad (12)$$

$$f_{i,j} \leq M y_{i,j} \quad \forall (i,j) \in E \quad (13)$$

$$\sum_{(i,j) \in E} c_{i,j} y_{i,j} \leq SP \times px \quad (14)$$

$$\frac{u_{i,j}}{c_{i,j}} + \frac{k\hat{u}_{i,j}}{c_{i,j}} \leq \max \quad \forall (i,j) \in E \quad (15)$$

$$0 \leq f_{i,j} \leq M \quad \forall (i,j) \in E \quad (16)$$

$$y_{i,j} \in \{0, 1\} \quad \forall (i,j) \in E \quad (17)$$

$$-M \leq v_{i,j} \leq M \quad \forall (i,j) \in E \quad (18)$$

$$0 \leq u_{i,j} \leq t_{i,j} \quad \forall (i,j) \in E : t_{i,j} \geq 0 \quad (19)$$

$$u_{i,j} = 0 \quad \forall (i,j) \in E : t_{i,j} < 0 \quad (20)$$

$$0 \leq \hat{u}_{i,j} \leq M \quad \forall (i,j) \in E \quad (21)$$

The objective function (1) minimizes the cyclist's effort. The set of constraints (2) is used to associate the energetic cost of traveling the arc  $(i, j)$  to the use of human and electric energy. The sets of constraints (3–6) are the flow balance constraints. The sets of constraints (7–12) are used to impose that the variables  $y$  define a path from  $s$  to  $p$ . The set of constraints (13) are used to associate the flow on the generic arc  $f_{i,j}$  to the use of the arc  $y_{i,j}$ . The constraint (14) is used to limit the cost of the path associated with the solution. The constraint (15) is used to limit the maximum effort of the cyclist to the parameter  $\max$ .

### 3 Computational Results

In this section we present our computational results. All the experiments were performed on a OSX 10.9 operating system, 16 GB of RAM and a quad-core processor Intel I7 running at 2.6 GHz. The mathematical model was coded in Java and solved using IBM ILOG CPLEX 12.7.

### 3.1 Graph Instances

Each graph instances used to run the tests is related to a real city of the world. To get the graphs we used OpenStreetMap [10] data. We created directed graphs, taking in account the direction of the roads. The maps are projected on a 2D plane. Each unit on this 2D plane represents 1 Km. For each vertex, we have an x and y coordinate in this 2D plane. To each arc of the graph is associated its length on the real street network. All the graphs are strongly connected. For each vertex of the graph, we report also the altitude; all these data are provided by the Google Maps Elevation API. Each graph contains a set of points, each point represents the position of a building on the map. This additional set of information can be useful for several problems like the Close Enough Traveling Salesman Problem (CETSP) [5, 6]. For each instance, we indicate the index of the four vertices associated with the 4 corners of the map, and the index of the central vertex. For each arc  $(i, j) \in A$  we define:

$$slope_{i,j} = \frac{altitude(j) - altitude(i)}{c_{i,j}}$$

$$t_{i,j} = \begin{cases} \frac{slope_{i,j} + 0.02}{2} & slope_{i,j} \leq -0.02 \\ slope_{i,j} + 0.01 & slope_{i,j} \geq -0.01 \\ 0 & otherwise \end{cases}$$

For each arc  $(i, j) \in A$  we define the constant  $t_{i,j}$  representing the energy consumption associate to the arc in relation to its slope. Of course, this parameter would depend on the vehicle.

### 3.2 Graph Reduction

In order to reduce the huge size of the input instances, we use the Dijkstra algorithm [9]. We compute the shortest path  $SP$  from  $s$  to  $p$ , and we remove from the input graph all the vertices  $v \in V$  such that the shortest path from  $s$  to  $v$  plus the shortest path from  $v$  to  $p$  is greater than  $px * SP$ . In such cases, going from  $v$  the path becomes the  $px\%$  more expensive than the shortest path. In Fig. 2 we show a graph with  $|V| = 18585$  and  $|A| = 36894$ , in red the shortest pat from  $s$  to  $p$ . Using the reduction technique ( $px = 20\%$ ), we obtain the graph in Fig. 3 with  $|V| = 4545$  and  $|A| = 9119$ , reducing the size of the graph of about 75%.





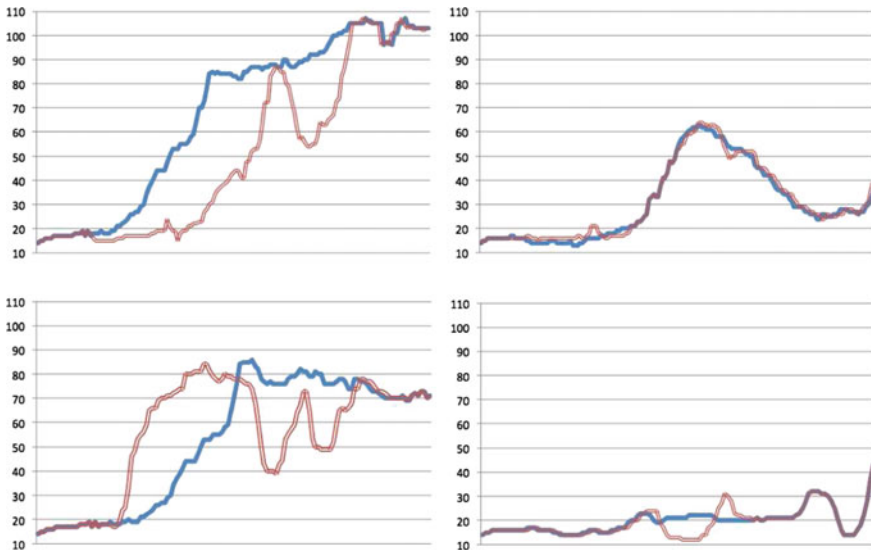
**Fig. 2** Map of Rome, in red the shortest path from the center to the north left corner

**Fig. 3** Map of Rome after the reduction technique, its size is decreased by 75%





**Fig. 4** Map of Rome. The altitude is shown by the background color (highest is brighter). The map shows eight paths starting from the center to the four corners. In red the shortest paths, in blue the paths produced by our MIP model



**Fig. 5** For each one of the four paths: Center—(Top Left, Top Right, Bottom left, Bottom Right), this Figure shows the elevation graph along the path, in red the shortest path in blue our path. The abscissa axis represents the elevation in meters, while the ordinate axis represents the path length

**Table 1** Computational results on the Rome's graph considering different destination nodes

Path from	Center	Path from	Center
Path to	Top left	Path to	Top right
MIP		MIP	
Cost	9.27 km	Cost	7.84 km
Value	197.46	Value	121.42
$u$	190.89	$u$	120.22
$\hat{u}$	3.29	$\hat{u}$	0.6
Max	0.06	Max	0.06
Running time	21 s	Running time	39 s
Shortest path		Shortest Path	
Cost	8.07 km	Cost	7.57 km
Value	204.31	Value	125.05
Max	0.22	Max	0.1
Path from	Center	Path from	Center
Path to	Bottom left	Path to	Bottom right
MIP		MIP	
Cost	8.4 km	Cost	6.96 km
Value	160.02	Value	111.3
$u$	152.91	$u$	111.3
$\hat{u}$	3.55	$\hat{u}$	0
Max	0.06	Max	0.04
Running time	16 s	Running time	6 s
Shortest path		Shortest path	
Cost	7.96 km	Cost	6.68 km
Value	178.64	Value	118.01
Max	0.22	Max	0.06

### 3.3 Rome

We used the graph of Rome to perform the experiments reported in this section. We made this choice because Rome was built on seven hills; moreover Rome is crossed by a river, the Tiber. This morphology, rich of differences in altitude, makes it particularly challenging for the optimization of the electric bike path. The graph of Rome is characterized by  $|V| = 18585$ ,  $|A| = 36894$  and represents an area of  $68 \text{ km}^2$ . In Fig. 4 we graphically show the results of four experiments. We compute the paths starting from the center of the map to the four corners, in red we show the shortest paths, in blue the paths produced by our MIP model. We use as parameters configuration:  $M = 400$  sufficient for 40 km on a flat road;  $\hat{M} = 1$  equal to 1% of charge;  $max = 0.06$  is the maximum effort of the cyclist is sufficient to a slope of 5%;

$px = 20\%$  we are looking for a path at most 20% longer than the shortest path;  $k = 2$  we lost half of the energy in converting the kinetic energy into electrical energy. Table 1 shows our computational results. The running time is less the one minute for all the scenarios, the length of the shortest path is in the range (6.68–8.4) km. For each shortest path, we compute “Value” that represents the effort of the cyclist (in cases of an empty battery) and “max” that represents the maximum effort of the cyclist. Figure 5 is very useful to understand the effectiveness of this model: it shows for each one of the four paths, the elevation graph along the path, in red the shortest path in blue our path. It is easy to see that our solution, with respect to the shortest path avoids the ups and downs along the way. Table 2, which shows further computational results, is useful to understand the effectiveness of our approach. For all the graphs, except for (\*), we used the same setting of parameters used in the previous experiments, for graph (\*) we set  $\hat{M} = 200$  because with  $\hat{M} = 1$  there is not a feasible solution.

## 4 Conclusions

In this paper we propose a flow formulation mathematical model to identify paths for bi-fuel vehicles. We focused our attention on electric bikes, designing paths able to balance the use of the electric motor and the effort of the human propulsion. To reduce the size of the input graph we develop an easy technique. The computational results show the effectiveness of our approach, that in few seconds is able to produce paths that reduce both the maximum and the total effort of the cyclist. Nevertheless, to compute a solution suitable for a bi-fuel vehicle with an autonomy greater than an electric bike, it is necessary to develop metaheuristics approaches able to elaborate bigger graphs e.g. Carousel Greedy [8], Tabu Search [4], Genetic Algorithm [7]. In future works, we are also interested in taking advantage of information on the traffic [3], in order to design paths which take into account the air quality breathed by the cyclist.

**Table 2** Computational results on real street network graphs. The running times and the costs are respectively shown in seconds and kilometers. The instances are available on ResearchGate (DOI:10.13140/RG.2.2.10184.93442)

Place	V	A	From	To	MIP			Shortest path			
					Cost	Value	Max	Time	Cost	Value	Max
Athens	35,522	75,369	Center	Top right	10.54	243.3	0.06	88	10.05	251.9	0.11
- Athens	35,522	75,369	Center	Top left	8.95	234.3	0.06	11	8.76	235.1	0.14
Athens	35,522	75,369	Center	Bot. left	12.13	111.5	0.06	52	10.13	106.4	0.19
Athens (*)	35,522	75,369	Center	Bot. right	9.32	520.2	0.05	8	9.15	524.8	0.29
Malta	27,786	60,440	Center	Bot. right	14.57	198.8	0.06	110	13.87	212.2	0.11
Washington D.C.	27,766	61,743	Center	Bot. right	11.91	217.2	0.06	14	11.64	240.4	0.19
Singapore	72,777	133,831	Center	Bot. right	33.13	386.35	0.06	253	32.81	386.5	0.11

## References

1. Baum, M., Dibbelt, J., Pajor, T., Wagner, D.: Energy-optimal routes for electric vehicles. In: Proceedings of the 21st ACM SIGSPATIAL international conference on advances in geographic information systems, pp. 54–63 (2013)
2. Baum, M., Dibbelt, J., Hbschle-Schneider, L., Pajor, T., Wagner, D.: Speed-consumption trade-off for electric vehicle route planning. In: OASIS-OpenAccess Series in Informatics, vol. 42 (2014)
3. Bianco, L., Cerrone, C., Cerulli, R., Gentili, M.: Locating sensors to observe network arc flows: exact and heuristic approaches. *Comput. Oper. Res.* **46**, 12–22 (2014)
4. Carrabs, F., Cerrone, C., Cerulli, R.: A Tabu search approach for the circle packing problem. In: 17th International Conference on IEEE Network-Based Information Systems (NBIS), pp. 165–171 (2014)
5. Carrabs, F., Cerrone, C., Cerulli, R., Gaudioso, M.: A novel discretization scheme for the close enough traveling salesman problem. *Comput. Oper. Res.* **78**, 163–171 (2017)
6. Carrabs, F., Cerrone, C., Cerulli, R., D’Ambrosio, C.: Improved upper and lower bounds for the close enough traveling salesman problem. In: International Conference on Green, Pervasive, and Cloud Computing, pp. 165–177. Springer, Cham (2017)
7. Cerrone, C., Cerulli, R., Gaudioso, M.: OMEGA one multi ethnic genetic approach. *Optim. Lett.* **10**(2), 309–324 (2016)
8. Cerrone, C., Cerulli, R., Golden, B.: Carousel greedy: a generalized greedy algorithm with applications in optimization. *Comput. Oper. Res.* **85**, 97–112 (2017)
9. Dijkstra, E.W.: A note on two problems in connexion with graphs. *Numer. Math.* 269–271 (1959)
10. OpenStreetMap contributors: Planet dump retrieved from <http://planet.osm.org>. <http://www.openstreetmap.org> (2017)
11. Sachenbacher, M., Leucker, M., Artmeier, A., Haselmayr, J.: Efficient energy-optimal routing for electric vehicles. In: AAAI (2011)
12. Salimifard, K., Raeesi, R.: A green routing problem: optimising CO<sub>2</sub> emissions and costs from a bi-fuel vehicle fleet. *International Journal of Advanced Operations Management* **6**, 27–57 (2014)
13. Zündorf, T.: Electric vehicle routing with realistic recharging models. Doctoral dissertation (2014)

# On the Forward Shortest Path Tour Problem

Francesco Carrabs, Raffaele Cerulli, Paola Festa and Federica Laureana

**Abstract** This paper addresses the Forward Shortest Path Tour Problem (FSPTP). Given a weighted directed graph, whose nodes are partitioned into clusters, the FSPTP consists of finding a shortest path from a source node to a destination node and which crosses all the clusters in a fixed order. We propose a polynomial time algorithm to solve the problem and show that our algorithm can be easily adapted to solve the shortest path tour problem, a slightly different variant of the FSPTP. Moreover, we carried out some preliminary computational tests to verify how the performance of the algorithm is affected by parameters of the instances.

**Keywords** Shortest path tour · Polynomial algorithm · Electric vehicles

## 1 Introduction

Given a directed graph  $G = (V, A)$ , where  $V$  is the set of nodes partitioned into  $T_1, \dots, T_N$  pairwise disjoint subsets, and  $A$  is the set of arcs, and given a cost function  $c$  that associates a nonnegative cost  $c(i, j)$  to each arc  $(i, j) \in A$ , the FSPTP consists of finding a shortest path from a source node  $s$  to a destination node  $d$  in  $G$  such that:

---

F. Carrabs · R. Cerulli · F. Laureana (✉)

Department of Mathematics, University of Salerno, Via Giovanni Paolo II, 132,  
84084 Fisciano (SA), Italy  
e-mail: flaureana@unisa.it

F. Carrabs  
e-mail: fcarrabs@unisa.it

R. Cerulli  
e-mail: raffaele@unisa.it

P. Festa  
Department of Mathematics and Applications, University of Napoli Federico II,  
Compl. MSA, Via Cintia, 80126 Napoli, Italy  
e-mail: paola.festa@unina.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_53

- (i) at least one node of each cluster is visited;
- (ii) it is possible to visit a node in  $T_k$  if and only if at least a node of each previous cluster,  $T_1, \dots, T_{k-1}$ , has been already visited.

The cost  $c(p)$  of a forward path tour  $p$  is given by the sum of the costs of the arcs it crosses.

A small instance of the problem is depicted in Fig. 1, where  $N = 4$ ,  $T_1 = \{s = 1\}$ ,  $T_2 = \{3\}$ ,  $T_3 = \{2\}$ , and  $T_4 = \{d = 4\}$ . The optimal forward path tour from 1 to 4 is  $p^* = \{1, 3, 2, 3, 4\}$ , and  $c(p^*) = 16$ .

This problem was introduced by Bertsekas in Dynamic Programming and Optimal Control [1], stating that it could be solved in polynomial time.

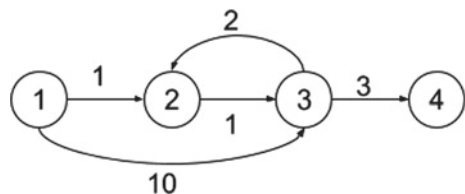
It arises in several heterogeneous contexts, including travel planning of the electric cars, where the low autonomy of this kind of vehicles requires an appropriate planning of the charging station stops along the travel (see [2]). In more detail, let us consider the problem to go from a source point  $s$  to a destination point  $d$  by using an electric vehicle, whose battery has a range of  $t$  kilometers. The aim is to find the shortest path from  $s$  to  $d$ , organizing the stops at the charging stations placed along the route. This problem can be modelled through a directed graph, whose nodes are the starting point  $s$ , the destination point  $d$ , and all the available charging stations, while the arcs represent the routes among the charging stations and their costs represent the distance in kilometers. Moreover, as shown in Fig. 2, the nodes are partitioned in clusters according to their distance from  $s$ . W.l.o.g. we assume that the first cluster  $T_1$  contains the only source node  $s$  and the last cluster  $T_N$  contains the only destination node  $d$ . Then, the second cluster  $T_2$  contains the charging stations whose distance from  $s$  is at most equal to  $t$  kilometers;  $T_3$  contains those stations placed on a distance from  $s$  variable from  $t$  to  $2t$  kilometers, and so on.

Another possible application is related to planning routes for a flying drone in which we need to cover, in a fixed order, a given set of targets to get a map of the monitored territory (see [3, 4]).

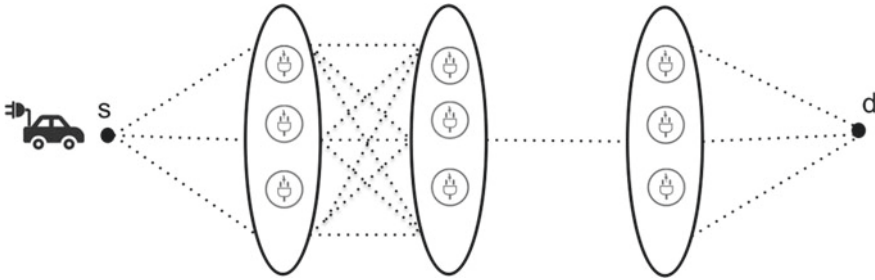
A very similar problem, named Shortest Path Tour problem (SPTP), was introduced in [5]. The aim of the SPTP is to obtain a shortest path from  $s$  to  $d$  in  $G$ , that successively passes through  $T_1, \dots, T_N$ . However, moving from the cluster  $T_{k-1}$  to the cluster  $T_k$ , a feasible path can cross any other node of the graph, even nodes in clusters  $T_h$ , with  $h > k$ .

If we consider again the graph in Fig. 1, the optimal solution of this problem is the path  $p' = \{1, 2, 3, 2, 3, 4\}$ , and its cost is  $c(p') = 8$ .

**Fig. 1** Graph  $G = (V, A)$ , with  $T_1 = \{s = 1\}$ ,  $T_2 = \{3\}$ ,  $T_3 = \{2\}$ , and  $T_4 = \{d = 4\}$







**Fig. 2** An electric car going from  $s$  to  $d$ , passing through the charging stations

In [5], Festa proved that the SPTP belongs to the class P, since it can be polynomially reduced to a classical shortest path problem on a suitably built multi-stage graph  $G'$ . However, due to the size of  $G'$ , the running times of the proposed solution approaches are acceptable only for small instances. A more effective approach for the SPTP was introduced in [6].

In this paper, we introduce a polynomial time algorithm to solve the FSPTP, based on several calls of Dijkstra’s algorithm. Our proposal can be trivially adapted to solve the SPTP by exhibiting the same computational complexity of the best known solution approach for the SPTP described in [6].

The remainder of this paper is organized as follows. Section 2 describes the polynomial time algorithm for the FSPTP and reports the proof of its correctness. Section 3 provides an example of how the algorithm works on a small instance. Computational results are reported in Sect. 4, and finally Sect. 5 concludes with some final remarks.

## 2 A Polynomial Time Algorithm for the FSPTP

Let  $G_k, k = 1, \dots, N$ , be the subgraph of  $G$  induced by  $\bigcup_{i=1}^k T_i$ . Let  $P_k, k = 2, \dots, N$ , be the set of the shortest paths from every node in  $T_{k-1}$  to any node in  $T_k$  computed on the graph  $G_k$ . Furthermore, let  $P_k^s, k = 2, \dots, N$  be the set of the shortest path tours from  $s$  to any node in  $T_k$ . Note that,  $P_2^s$  is exactly the set of the shortest paths from  $s$  to any node in  $T_2$  on the graph  $G_2$ .

The following lemma holds:

**Lemma 1** *Let  $p_k$  be the forward shortest path tour from  $s$  to a node  $y \in T_k$ . Then,  $p_k$  is obtained by concatenating a forward shortest path tour belonging to  $P_{k-1}^s$ , with a path belonging to  $P_k$  ending in  $y$ .*

*Proof* Since  $p_k$  is a forward shortest path tour, there exists at least a node in  $p_k$  belonging to  $T_{k-1}$ . W.l.o.g. let us suppose that  $x$  is the last node of  $T_{k-1}$  belonging to  $p_k$ . It is possible to decompose  $p_k$  in two subpaths:  $p_{k-1}$ , from  $s$  to  $x$ , and  $p_{xy}$  from  $x$  to  $y$ . Therefore, by definition of forward shortest path tour, it results that  $p_{k-1} \in P_{k-1}^s$  and  $p_{xy} \in P_k$ . □

Based on the previous theoretical result, starting from  $s \in T_1$ , algorithm A1 sequentially computes  $p_k \in P_k^s, k = 2, \dots, N$  as described in the following.

**Initialization:** Compute  $P_2^s$  by applying any shortest path algorithm (Dijkstra [7], Auction [8], [9], and so on)

**For each  $k = 3$  to  $N$ :** Let  $y$  be a node in  $T_k$ . For any  $x \in T_{k-1}$ , compute the cost of the forward shortest path tour from  $s$  to  $x$ , plus the cost of the path from  $x$  to  $y$  belonging to  $P_k$ . Among the  $|T_{k-1}|$  paths computed, the algorithm selects the shortest one as the forward shortest path tour from  $s$  to  $y$ . The procedure is repeated for each  $y \in T_k$ , by generating in this way  $P_k^s$ .

The following theorem establishes the correctness of the algorithm.

**Theorem 1** *Algorithm A1 finds the forward shortest path tour from  $s$  to any other node in the graph  $G$ .*

*Proof* Let  $y$  be any node in  $T_k, k = 2, \dots, N$ . To compute the forward shortest path tour  $p$  from  $s$  to  $y$ , the algorithm concatenates an element  $p_1$  in  $P_{k-1}^s$  and an element  $p_2$  in  $P_k$ , such that  $c(p_1) + c(p_2)$  is minimum. Suppose by contradiction that  $p$  is not the optimum, so there exists a forward shortest path tour  $p'$  for which  $c(p') < c(p)$ . From Lemma 1,  $p'$  can be written as the concatenation of a forward path tour  $p'_1$  in  $P_{k-1}^s$  and a path  $p'_2$  in  $P_k$ . Therefore, it results that  $c(p'_1) + c(p'_2) < c(p_1) + c(p_2)$ , but this is a contradiction because the algorithm chooses the pair  $(p_1, p_2)$  that minimizes the sum of the costs. □

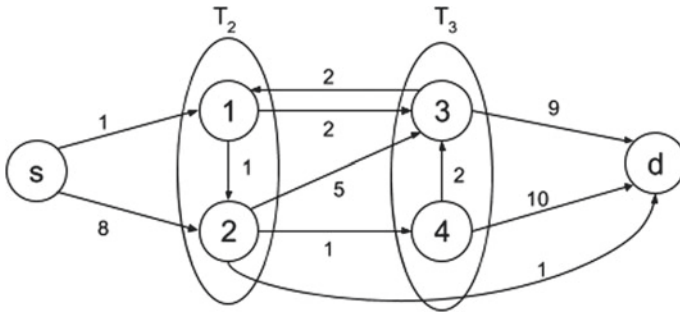
### 3 Example

The following example allows to clarify the behaviour of the algorithm.

Let us consider the graph in Fig. 3, with  $V = \{s, 1, 2, 3, 4, d\}, T_1 = \{s\}, T_2 = \{1, 2\}, T_3 = \{3, 4\},$  and  $T_4 = \{d\}$ .

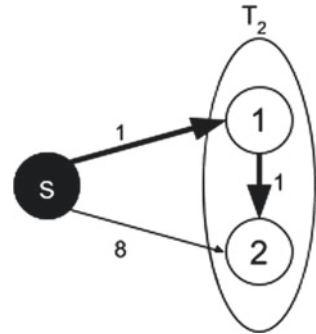
The steps of the algorithm A1 are:

- **Step 1:** the algorithm computes  $P_2^s$ , i.e. the shortest paths from  $s$  to any other node in  $T_2$  on the graph  $G_2$ . Therefore, as it is shown in Fig. 4,  $P_2^s = \{\{s, 1\}, \{s, 1, 2\}\}$ .
- **Step 2:** the algorithm computes  $P_3$ . As it can be seen in Fig. 5a, the shortest path from 1 to 3 on  $G_3$  is  $\{1, 3\}$ , and the shortest path from 2 to 3 on  $G_3$  is  $\{2, 4, 3\}$  (Fig. 5b). So  $P_3 = \{\{1, 3\}, \{2, 4, 3\}, \{2, 4\}, \{1, 2, 4\}\}$ . The forward shortest path tour from  $s$  to 3 is the minimum between the two paths obtained through the concatenation of an element of  $P_2^s$  and an element of  $P_3$ , so the possibilities



**Fig. 3** Graph  $G = (V, A)$  with  $N = 4$ ,  $T_1 = \{s\}$ ,  $T_2 = \{1, 2\}$ ,  $T_3 = \{3, 4\}$ ,  $T_4 = \{d\}$

**Fig. 4** Shortest paths from  $s$  to each node in  $T_2$  on the graph  $G_2$

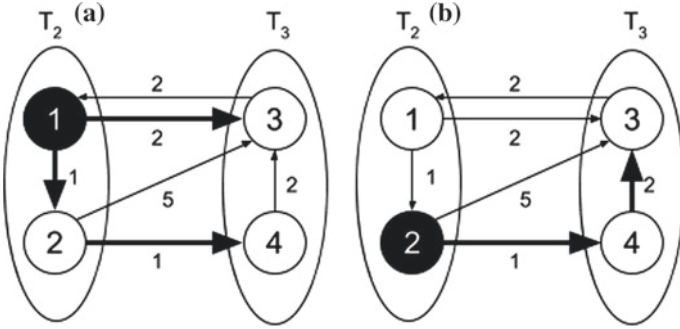


are:  $\{s, 1, 3\}$ , whose cost is 3, or  $\{s, 1, 2, 4, 3\}$ , whose cost is 5. Thus the forward shortest path tour from  $s$  to 3 is  $\{s, 1, 3\}$ . In the same way we can see that the forward shortest path tour from  $s$  to 4 is  $\{s, 1, 2, 4\}$ , whose cost is 3. Thus  $P_3^s = \{\{s, 1, 3\}, \{s, 1, 2, 4\}\}$ .

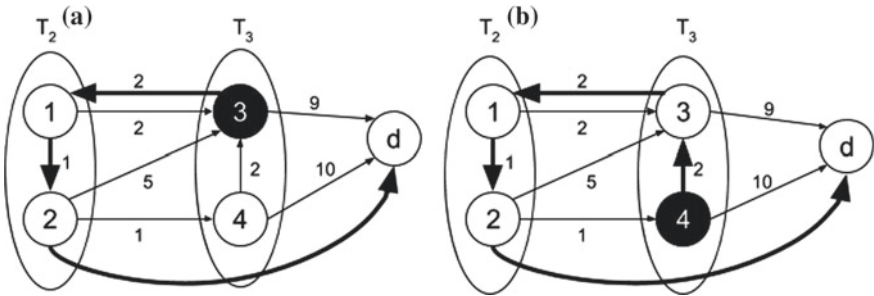
- **Step 3:** the algorithm computes  $P_4 = \{\{3, 1, 2, d\}, \{4, 3, 1, 2, d\}\}$ , as it can be seen in Fig. 6a, b. Proceeding in the same way as the previous step, i.e. through concatenation, we obtain that the forward shortest path tour from  $s$  to  $d$  is  $\{s, 1, 3, 1, 2, d\}$ , whose cost is 7, as it is shown in Fig. 7.

A nice property of algorithm A1 is that it can be trivially adapted for the resolution of the SPTP proposed by Festa. Indeed it suffices to consider at each step the original graph, without deleting the arcs between clusters not already visited. Therefore, when we compute the shortest paths between the clusters  $T_k$  and  $T_{k+1}$ , we can eventually pass through nodes belonging to  $T_h$ , with  $h > k$ .

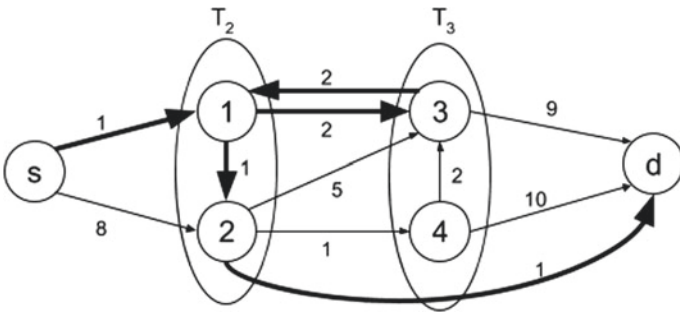
This version of the algorithm is equivalent, from a computational point of view, to the one proposed in [6].



**Fig. 5** **a** Shortest paths from 1 to each node in  $T_3$  on  $G_3$ . **b** Shortest paths from 2 to each node in  $T_3$  on  $G_3$



**Fig. 6** **a** Shortest path from 3 to  $d$ . **b** Shortest path from 4 to  $d$



**Fig. 7** The forward shortest path tour from  $s$  to  $d$  is the path  $\{s, 1, 3, 1, 2, d\}$ , whose cost is 7

## 4 Computational Results

In this section, we evaluate the performance of A1 depending on the number of nodes, the density of the graph, and the number of clusters. Our algorithm was coded in C++ on an OSX platform, running on an Intel Core i7 3.4 GHz processor with 8 GB of RAM.

Since the computational time of A1 was lower than a second, and then negligible, on instances up to 500 nodes, we randomly generated a set of instances containing at least 1000 nodes. In more detail, our instances were generated as follows. Chosen the number of nodes  $n$ , the number of arcs  $m$ , and the number of clusters  $N$ , the source

**Table 1** Computational results on randomly generated instances with 1000, 1500 and 2000 nodes

n	m	N	Time
1000	499500	50	10.51
1000	499500	150	3.23
1000	499500	250	2.10
1000	719280	50	15.02
1000	719280	150	4.82
1000	719280	250	2.93
1000	999000	50	18.26
1000	999000	150	6.38
1000	999000	250	3.70
1500	1124250	75	34.55
1500	1124250	225	11.16
1500	1124250	375	6.11
1500	1618920	75	47.27
1500	1618920	225	15.66
1500	1618920	375	9.73
1500	2248500	75	63.10
1500	2248500	225	21.36
1500	2248500	375	12.48
2000	1999000	100	77.65
2000	1999000	300	26.95
2000	1999000	500	15.34
2000	2878560	100	111.09
2000	2878560	300	36.99
2000	2878560	500	22.34
2000	3998000	100	141.21
2000	3998000	300	48.13
2000	3998000	500	29.31

node  $s$  and the destination node  $d$  are randomly selected, as well as the partition of  $V$  into  $N$  clusters. Obviously, in each cluster there must be at least one node. Finally, the arcs are randomly added, ensuring that at least a feasible solution there exists. The parameters used to generate the instances are:  $n \in \{1000, 1500, 2000\}$ ,  $m \in \{0.5n(n-1), 0.75n(n-1), n(n-1)\}$ , and finally  $N \in \{0.05n, 0.15n, 0.25n\}$ . The arc costs are randomly chosen in the range  $[25, 100]$ .

These instances have a high arc density, because they better reflect the real application of the problem. Moreover, our algorithm solves instances with low arc density in a short time, less than a minute, while the instances with high arc density result harder to solve, with a computational time that is even an order of magnitude greater than the one of the low arc density.

Computational results are shown in Table 1, whose columns report the number of nodes, the number of arcs, the number of clusters, and the CPU time (in seconds), respectively.

It is evident from these results that the number of clusters  $N$  is the parameter that mostly affects the performance of A1. In particular, as the value of  $N$  increases, the CPU time decreases. For instance, by considering the last three lines of the table, we observe that with  $N = 100$  the CPU time is equal to 141.21, while for  $N = 500$  the CPU time is equal to 29.31, thus resulting in a reduction of 80% of the computational time. The reason behind this behaviour is that the higher is the number of clusters, the smaller is their cardinality, and the lower is the number of possible choices that A1 can perform. Indeed, in the extreme case when  $n = N$ , we have exactly a single path from  $s$  to  $d$  and, obviously, the algorithm finds it instantly.

## 5 Conclusions

We have proposed a polynomial algorithm for the FSPTP, a variant of the classical shortest path problem. The algorithm is based on the resolution of a sequence of shortest path problems. We also have exhibited a proof of its correctness and an example of how it works on a small instance. The computational results show that the number of clusters heavily affects the performance of the algorithm that, in any case, is able to solve in less than a minute almost all the instances.

Moreover, our algorithm can be used to solve also a slightly different problem, the SPTP, introduced by Festa in [5].

Possible future investigation could be done comparing the adaptation of our proposal for solving the SPTP, with the best known algorithm for that similar problem, and more deeply analyzing how the performance of the algorithm varies depending on the number of nodes and arcs, the number of clusters and their size. We expect that the relationship between the behavior of our algorithm and the number of clusters, shown by the computational results, is not due to the specific instances used in the tests, but is an intrinsic characteristic of the problem.

Moreover, if the size of the instance we have to solve is too big, a possible approach is to split the graph in smaller subgraphs, using a graph partitioning algorithm (see [10]).

## References

1. Bertsekas, D.P.: *Dynamic Programming and Optimal Control*, vol. I, 3rd edn. Athena Scientific (2005)
2. Artmeier, A., Haselmayr, J., Leucker, M., Sachenbacher, M.: The shortest path problem revisited: optimal routing for electric vehicles. In: *Proceedings of the KI 2010, Advances in Artificial Intelligence, 33rd Annual German Conference on AI, Karlsruhe, Germany*, pp. 309–316, 21–24 Sept 2010. Springer, Berlin, Heidelberg
3. Carrabs, F., Cerrone, C., Cerulli, R., Gaudioso, M.: A novel discretization scheme for the close enough traveling salesman problem. *Computers & OR* **78**, 163–171 (2017)
4. Carrabs, F., Cerulli, R., Cerrone, C., D’Ambrosio, C.: Improved upper and lower bounds for the close enough traveling salesman problem, Green, Pervasive, and Cloud Computing. In: *Proceedings of the 12th International Conference, GPC 2017, Cetara, Italy*, pp. 165–177, 11–14 May 2017. Springer International Publishing (2017)
5. Festa, P.: Complexity analysis and optimization of the shortest path tour problem. *Optim. Lett.* **6**(1), 163–175 (2012)
6. Festa, P., Guerriero, F., Laganà, D., Musmanno, R.: Solving the shortest path tour problem. *Eur. J. Oper. Res.* **230**, 464–474 (2013)
7. Dijkstra, E.: A note on two problems in connexion with graphs. *Numer. Math.* **1**, 269–271 (1959)
8. Bertsekas, D.P.: An auction algorithm for shortest paths. *SIAM J. Optim.* **1**, 425–447 (1991)
9. Bertsekas, D.P., Pallottino, S., Scutellà, M.G.: Polynomial auction algorithms for shortest paths. *Comput. Optim. Appl.* **4**, 99–125 (1995)
10. Lum, O., Cerrone, C., Golden, B., Wasil, E.: Partitioning a street network into compact, balanced, and visually appealing routes. *Networks* (2017)

# A Flow Formulation for the Close-Enough Arc Routing Problem

Carmine Cerrone, Raffaele Cerulli, Bruce Golden and Rosa Pentangelo

**Abstract** The close-enough arc routing problem is a generalization of the classic arc routing problem and it has many interesting real-life applications. In this paper, we propose some techniques to reduce the size of the input graph and a new effective mixed integer programming formulation for the problem. Our experiments on directed graphs show the effectiveness of our reduction techniques. Computational results obtained by comparing our MIP model with the existing exact methods show that our algorithm is really effective in practice.

**Keywords** Close enough arc routing problem · MIP model · Vertex cover

## 1 Introduction

The Close-Enough Arc Routing Problem (CEARP) is a generalization of the Rural Postman Problem (RPP). Let  $G = (V, A, M)$  be a directed graph with a set of vertices  $V$ , a set of arcs  $A$ , and a set of targets  $M$  located on arcs. An arc  $a \in A$  covers a target  $m \in M$  iff the target is either on the arc or within a predetermined distance (radius) from the arc. Let  $N = \{(m, a) | m \in M, a \in A\}$  be a set containing the couple (target  $m$ , arc  $a$ ) if and only if the arc  $a$  covers the target  $m$ , and let  $c_{ij}$  be the cost associated with arc  $a = (i, j) \in A$ . Finally, let  $v_0 \in V$  indicate the depot node. The

---

C. Cerrone

Department of Biosciences and Territory, University of Molise, Campobasso, Italy  
e-mail: carmine.cerrone@unimol.it

R. Cerulli · R. Pentangelo (✉)

Department of Mathematics, University of Salerno, Fisciano, Italy  
e-mail: rpentangelo@unisa.it

R. Cerulli

e-mail: raffaele@unisa.it

B. Golden

Robert H. Smith School of Business, University of Maryland, College Park, USA  
e-mail: bgolden@rhsmith.umd.edu

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_54



*CEARP* consists of finding a minimum cost tour starting and ending at the depot node  $v_0$ , traversing a subset of arcs such that all the targets in  $M$  are covered. The *CEARP* problem was introduced by Drexl [6, 7], he proved that the problem is NP-hard, and he proposed a branch-and-cut algorithm. Shuttleworth et al. [11] proposed four heuristics to solve instances with approximately 9000 arcs. A mixed integer programming (MIP) formulation for the problem was introduced by Há et al. [8]; the same authors presented a new IP formulation in [9]. Ávila et al. [1] proposed a branch-and-cut algorithm which they compared with the model presented in [9], providing good computational results. In [10], Lum et al. propose some techniques to partition a graph in order to simplify its structure.

There are several real-life applications for this problem. The meter reading problem is an important application of the *CEARP*: A vehicle with a receiver on board travels over a street network. If it traverses a street and is closer than a certain distance to a RFID meter (e.g., in a home), the receiver is able to read the value of the meter. An interesting variant of this real-life problem is when flying drones are used to read the meters [3].

The remainder of this paper is organized as follows. In Sect. 2, we present our new MIP formulation. In Sect. 3, we show the computational results of our approach, comparing them with recent results proposed in [1, 9] and, finally, in Sect. 4, we present our conclusions.

## 2 Flow Formulation

In this section, we propose a very effective mathematical programming formulation for the *CEARP* based on an efficient graph reduction procedure. Moreover, we show how, using any vertex cover computed on the input graph, we can take advantage of its features to identify a set of important vertices for the routing problems.

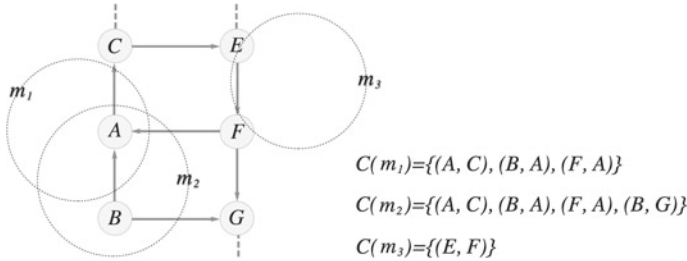
### 2.1 Graph Reduction

In order to improve the resolution process, we prove some properties of the problem and we provide some definitions that will help us to reduce the size of input instances.

**Definition 1** For each  $m \in M$ ,  $C(m) = \{a \in A | (m, a) \in N\}$  is the set of all the arcs that cover the target  $m$ .

It is possible to reduce the number of targets, taking into consideration the following property:

**Property 1** Let  $m_1, m_2 \in M$  be a couple of targets, the target  $m_2$  is redundant if  $C(m_1) \subseteq C(m_2)$ .



**Fig. 1** Redundant target  $m_2$ . Necessary vertex A. Necessary arc  $(E, F)$

*Proof* Let  $G' = (V, A, M')$  be the graph where  $M' = M \setminus \{m_2\}$ . Each tour  $T'$  in  $G'$  covering each target in  $M'$  contains at last one arc  $a \in C(m_1)$  and then the tour  $T'$  is also a feasible tour for the graph  $G = (V, A, M)$  (see Fig. 1).

In the following, we give two definitions to characterize the set of vertices and the set of arcs that are necessary in every feasible solution.

**Definition 2** The set  $\hat{V} = \{v \in V \mid \exists m \in M \text{ such that } v = i \text{ or } v = j, \forall (i, j) \in C(m)\}$  is the set of necessary vertices.

**Definition 3** The set  $\hat{A} = \{a \in A \mid \exists m \in M : C(m) = \{a\}\}$  is the set of necessary arcs.

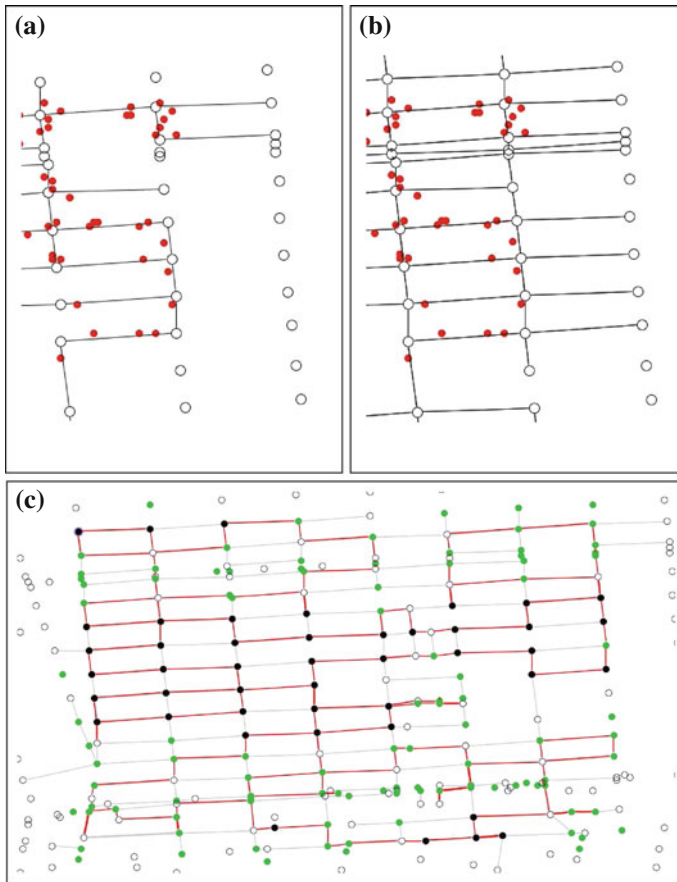
Finally, by using the following two properties, we can try to reduce the size of the input instances.

**Property 2** The target  $m \in M$  is redundant if  $\exists a \in \hat{A}$  such that  $a \in C(m)$ .

**Property 3** The target  $m \in M$  is redundant if  $\exists (i, j) \in C(m)$  such that  $i \in \hat{V}$  or  $j \in \hat{V}$ .

### 2.2 The Vertex Cover

In this section, to strengthen the MIP model, we will use some information obtained from solving a vertex cover problem related to the directed graph  $G = (V, A, M)$  defining our problem. We consider the graph  $G' = (V, E, M)$  obtained from  $G$  by transforming each directed arc into an undirected edge and leaving in  $E$  only edges associated with arcs of  $G$  that cover at least one target in  $M$ . On  $G'$ , we solve a vertex cover problem getting a subset  $(VC)$  of its vertices such that each edge of  $G'$  is incident to at least one vertex of  $VC$ . In Fig. 2a, we show a portion of a real street network. The red dots represent the targets that we need to cover and the edges are those in which at least one target is located. In Fig. 2b are drawn all edges in which



**Fig. 2** **a** Edges in which there is at least one target (red dot). **b** All edges in which it is possible to read at least one target. **c** A feasible solution

it is possible to cover at least one target (we construct  $G'$  considering only these edges). By computing the set  $VC \subseteq V$  as a vertex cover obtained using the edges of Fig. 2b, we get a set of vertices  $VC$  from which we can efficiently check if a set of arcs on  $G$ , necessary to cover all the targets in  $M$ , is connected for each feasible solution. Obviously the vertex cover with the minimum cardinality would be the best possible set  $VC$ , but, in many cases, it is sufficient to compute just a feasible solution using an heuristic procedure [5] or a metaheuristic approach like tabu search [2] or a genetic algorithm [4]. Heuristically, we can compute  $VC$  using a simple two-step procedure. In step 1, we insert in  $VC$  all the vertices in  $\hat{V}$ . In step 2, we add to  $VC$  all the other vertices necessary to get a feasible solution. In Fig. 2c, we report, in black, the vertices in the set  $\hat{V}$  and, in green, the vertices in the set  $VC \setminus \hat{V}$ . If we look

at the edges as bi-directed arcs, the red edges represent a feasible solution  $T$  for the CEARP. Indeed each edge in  $T$  is incident to at least a green or a black vertex.

### 2.3 MIP Model

In this section, we describe our MIP model based on a flow formulation. From now on, the depot will be vertex 0. In order to proceed with the description of the model, we define the variables that will be used. For each arc  $(i, j) \in A$ , we have:

- $x_{ij} \in Z_0^+$  is the number of times the arc  $(i, j)$  is traversed.
- $f_{ij} \in R_0^+$  are the flow variables associated with arc  $(i, j)$ .

The model can be formulated as follows:

$$\text{Minimize } \sum_{(i,j) \in A} c_{ij} x_{ij} \tag{1}$$

$$\sum_{a=(i,j) \in A \setminus \{(m,a) \in N\}} x_{ij} \geq 1 \quad \forall m \in M \tag{2}$$

$$x_{ij} \geq 1 \quad \forall (i, j) \in \hat{A} \tag{3}$$

$$\sum_{j \in V \setminus \{(i,j) \in A\}} x_{ij} - \sum_{j \in V \setminus \{(j,i) \in A\}} x_{ji} = 0 \quad \forall i \in V \tag{4}$$

$$\sum_{(0,j) \in A} f_{0j} \geq 1 \tag{5}$$

$$\sum_{j \in V \setminus \{(i,j) \in A\}} f_{ij} - \sum_{j \in V \setminus \{(j,i) \in A\}} f_{ji} = 0 \quad \forall i \in V \setminus \{VC \cup \{0\}\} \tag{6}$$

$$\sum_{j \in V \setminus \{(i,j) \in A\}} f_{ij} - \sum_{j \in V \setminus \{(j,i) \in A\}} f_{ji} = 1 \quad \forall i \in \hat{V} \setminus \{0\} \tag{7}$$

$$\sum_{j \in V \setminus \{(i,j) \in A\}} f_{ij} - \sum_{j \in V \setminus \{(j,i) \in A\}} f_{ji} = \sum_{j \in V \setminus \{(i,j) \in A\}} x_{ij} \quad \forall i \in VC \setminus \{\hat{V} \cup \{0\}\} \tag{8}$$

$$f_{ij} \leq Mx_{ij} \quad \forall (i, j) \in A \tag{9}$$

$$x_{ij} \in Z_0^+ \quad \forall (i, j) \in A \tag{10}$$

$$f_{ij} \in R_0^+ \quad \forall (i, j) \in A \tag{11}$$

The set of constraints (2) ensure that each target is covered at least from one arc of the solution and constraints (3) ensure that each necessary arc is in the solution. The set of constraints (4) ensure that for each node, the number of selected arcs in its forward star corresponds to the number of selected arcs in its backward star. The constraint (5) ensures outcoming flow from the depot. The set of constraints (6) set to 0 the amount of in-flow for all the vertices not necessary to guarantee the connection

**Table 1.** In both subtables, the first column shows the name of the instance, the remaining three columns show the objective function values. The star (\*) is associated with outliers (maybe depending on a different parsing of the instances). Optimal solutions are in bold

	Há et al.	Avilá et al.	Cerrone et al.	Há et al.	Avilá et al.	Cerrone et al.
1500_0_0.5	105431.4	<b>104892.6</b>	<b>104892.6</b>	1000_0_0.5	<b>76033.0</b>	<b>76033.0</b>
1500_1_0.5	97335.3	<b>96766.0</b>	96791.6	1000_1_0.5	<b>84237.2</b>	<b>84237.2</b>
1500_2_0.5	112429.6	<b>112102.1</b>	<b>112102.1</b>	1000_2_0.5	*89353.5	<b>89653.5</b>
1500_3_0.5	95378.1	<b>94897.8</b>	<b>94897.8</b>	1000_3_0.5	76384.4	<b>75954.9</b>
1500_4_0.5	102423.9	<b>101991.0</b>	<b>101991.0</b>	1000_4_0.5	85397.0	<b>85097.1</b>
1500_0_1	*129459.7	<b>129570.6</b>	<b>129570.6</b>	1000_0_1	*82387.1	<b>82687.1</b>
1500_1_1	123423.0	<b>123123.0</b>	<b>123123.0</b>	1000_1_1	*89396.5	<b>89896.5</b>
1500_2_1	<b>133418.3</b>	<b>133418.3</b>	<b>133418.3</b>	1000_2_1	98351.8	<b>98051.8</b>
1500_3_1	116458.8	<b>115943.8</b>	<b>115943.8</b>	1000_3_1	<b>82344.2</b>	<b>82344.2</b>
1500_4_1	117403.4	<b>116721.5</b>	<b>116721.5</b>	1000_4_1	*91315.6	<b>91915.6</b>
1500_0_5	162497.8	<b>162097.8</b>	<b>162097.8</b>	1000_0_5	100495.3	<b>100395.4</b>
1500_1_5	*160492.7	<b>160792.8</b>	<b>160792.8</b>	1000_1_5	109418.5	<b>109318.5</b>
1500_2_5	177442.4	<b>177242.4</b>	<b>177242.4</b>	1000_2_5	114462.3	<b>114362.3</b>
1500_3_5	*151452.9	<b>151852.9</b>	<b>151852.9</b>	1000_3_5	*103470.9	<b>103791.0</b>
1500_4_5	*161433.4	<b>161833.4</b>	<b>161833.4</b>	1000_4_5	*112425.0	<b>112625.0</b>
1500_0_10	*174404.1	<b>174504.1</b>	<b>174504.1</b>	1000_0_10	*110427.9	<b>110528.0</b>
1500_1_10	<b>173404.5</b>	<b>173404.5</b>	<b>173404.5</b>	1000_1_10	*113494.1	<b>113694.2</b>
1500_2_10	185430.8	<b>185330.8</b>	<b>185330.8</b>	1000_2_10	*123433.9	<b>126660.0</b>
1500_3_10	162471.7	<b>162071.7</b>	<b>162071.7</b>	1000_3_10	115481.4	<b>115281.4</b>
1500_4_10	*168434.3	<b>168734.3</b>	<b>168734.3</b>	1000_4_10	128463.2	<b>128163.3</b>

of the final solution. Constraints (7) set to 1 the amount of in-flow for all the vertices that we know will be traversed in any feasible solution. Constraints (8) defined for each vertex that could be used to ensure the connection of the solution, set an in-flow equal to the time that the vertex is traversed in the solution. Constraints (9) ensure that we can have flow only on the arcs used in the solution.

### 3 Computational Results

In this section, we present computational results obtained by using the MIP model. Our experiments were performed on a OSX 10.9 operating system, 16 GB of RAM and a quad-core processor Intel I7 running at 2.6 GHz. The MIP model was coded in Java and solved using IBM ILOG CPLEX 12.5.

The computational tests are performed on the set of benchmark instances presented in [8]. We compared our MIP model with the exact approaches proposed in [1, 9]. In Table 1 there are two sets of instances. In the first set, we have  $|V| = 500$  and  $|A| = 1500$  while, in the second one, we have  $|V| = 500$  and  $|A| = 1000$ . For each set, we have four different numbers of targets ( $\frac{1}{2}|A|$ ,  $|A|$ ,  $5|A|$ ,  $10|A|$ ).

Table 1 shows that our algorithm solves to optimality 39 instances. For these two sets, our approach is competitive with the approach of Avilá et al. and outperforms the results of Há et al. Table 2 shows that in terms of computational times our MIP model outperforms the previous approaches. In terms of numbers of solved instances, our approach is always better than Há et.al. [9]. For the sparse instances ( $|A| = 1000$ ), our model outperforms the results of Ávila et al. [1] with respect to computation times. For the dense instances ( $|A| = 1500$ ), Ávila et al. are able to find the optimal solution in 20 instances. Our approach finds the optimal solution in 19 instances; we

**Table 2** For the three models compared in this section, we show the running times (seconds) and the number of certified optimal solutions. Each row of the table shows the average value on five instances

	Há et al.		Avilá et al.		Cerrone et al.	
	Opt found	Time	Opt found	Time	Opt found	Time
1500_0.5	0	7202.9	5	830.5	1	874.0
1500_1	2	4499.9	5	1235.5	3	433.5
1500_5	5	154.5	5	49.2	5	14.6
1500_10	5	205.9	5	50.1	5	9.9
1000_0.5	3	4155.6	5	245.7	5	47.7
1000_1	4	2447.8	5	88.6	5	13.3
1000_5	5	315.1	5	28.3	5	6.0
1000_10	5	82.3	5	20.3	5	4.4

can certify optimality for 14 instances and for the remaining 6 instances, our gap is always less than the 1%.

## 4 Conclusions

We proposed a new MIP model for the CEARP based on a flow formulation and introduced some properties useful to reduce the size of the graph instances. For the benchmark instances, our graph reduction allowed to decrease the number of targets. This decrease ranges from 20 to 90% of the total number of them. The computational results show the effectiveness of our approach. For several instances, our approach is substantially faster than competing solution techniques.

## References

1. Ávila, T., Corberán, A., Plana, I., Sanchis, J.M.: A new branch-and-cut algorithm for the generalized directed rural postman problem. *Trans. Sci.* **50**(2), 750–761 (2016)
2. Carrabs, F., Cerrone, C., Cerulli, R.: A tabu search approach for the circle packing problem. In: 2014 17th International Conference on Network-Based Information Systems, pp. 165–171. IEEE (2014)
3. Carrabs, F., Cerrone, C., Cerulli, R., Gaudio, M.: A novel discretization scheme for the close enough traveling salesman problem. *Comput. Oper. Res.* **78**, 163–171 (2017)
4. Cerrone, C., Cerulli, R., Gaudio, M.: Omega one multi ethnic genetic approach. *Optim. Lett.* **10**(2), 309–324 (2016)
5. Cerrone, C., Cerulli, R., Golden, B.: Carousel greedy: a generalized greedy algorithm with applications in optimization. *Comput. Oper. Res.* **85**, 97–112 (2017)
6. Drexl, M.: On the generalized directed rural postman problem. *J. Oper. Res. Soc.* **65**(8), 1143–1154 (2014)
7. Drexl, M., Sebastian, H.-J.: On some generalized routing problems. Technical report, Deutsche Post Lehrstuhl für Optimierung von Distributionsnetzwerken (NN) (2007)
8. Há, M.H., Bostel, N., Langevin, A., Rousseau, L.-M.: An exact algorithm for the close enough traveling salesman problem with arc covering constraints. In: 1st International Conference on Operations Research and Enterprise Systems (ICORES), pp. 233–238. Portugal, Feb 2012
9. Há, H.M., Bostel, N., Langevin, A., Rousseau, L.-M.: Solving the close-enough arc routing problem. *Networks* **63**(1), 107–118 (2014)
10. Lum, O., Cerrone, C., Golden, B., Wasil, E.: Partitioning a street network into compact, balanced, and visually appealing routes. *Networks* **69**(3), 290–303 (2017)
11. Shuttleworth, R., Golden, B.L., Smith, S., Wasil, E.: *Advances in Meter Reading: Heuristic Solution of the Close Enough Traveling Salesman Problem over a Street Network*, pp. 487–501. Springer US, Boston, MA (2008)

# A Mesoscopic Approach to Model Route Choice in Emergency Conditions

Massimo Di Gangi and Antonio Polimeni

**Abstract** In this paper, a dynamic approach to simulate users' behaviour when a hazardous event occurs in a transport network is proposed. Particularly, a route choice model within a mesoscopic dynamic traffic assignment framework is described, assuming that users can acquire information on the network status in real time. The effects of the event are taken into consideration by introducing a risk factor in arc cost function in order to allow en-route changes in users' path choice decisions. The proposed approach is tested on a trial network, highlighting the evolution of changes in path choice caused by a hazardous event that modifies supply conditions.

**Keywords** Path choice • Evacuation • Dynamic traffic assignment

## 1 Introduction

A transport system involved in a hazardous event may suffer some modifications in terms both of demand (a large amount of users all want to move together) and of supply (the event can modify the transport system conditions, i.e. changes in capacity, unusable roads, etc.). Focusing on these aspects, it is necessary to model users' behaviour in the presence of changes in supply. To assist the users in their trips along the network, information must be provided either in real-time (independently of the position of the user in the network) or in preset points in the network. In both cases, information provided to the users may involve changes in path choice [7–9, 14, 15]. In design field, the paths are designed to minimize the evacuation time [12], to provide alternative paths [1], to optimize the movements of

---

M. Di Gangi (✉) · A. Polimeni  
Dipartimento di Ingegneria, Università degli Studi di Messina,  
C.da di Dio S. Agata, 98166 Messina, Italy  
e-mail: mdigangi@unime.it

A. Polimeni  
e-mail: antonio.polimeni@uniroma2.it



rescue vehicles [10]. A general architecture to simulate and design transportation system under evacuation conditions is proposed in [2]. Concerning assignment models to simulate evacuation, static approaches do not allow analysis of phenomena connected to temporal variations in terms of both demand and supply, such as rising and scattering of queues due to temporary peaks of demand and/or capacity reductions of infrastructures. Moreover, considering the emergency conditions, the transportation system cannot be studied considering an equilibrium approach. The Dynamic Traffic Assignment (DTA) model considered here to simulate evacuation is mesoscopic [5], where users may be assembled in packets that move on the network discretising demand for each origin-destination pair. The main contribution of the proposed model consists both in handling re-routing to simulate users' response to external sources of information (both on-board and external) and in taking into account of the risk within the arc cost function. The paper is structured as follows: Sect. 2 describes the formulation of the model; the proposed model is applied to a trial network in Sect. 3 where some specifications are itemised. In Sect. 4 major findings are recapped and some research perspectives are outlined.

## 2 Mesoscopic Model

The simulation is conducted for discrete time intervals assumed, for the sake of simplicity (but without prejudice to the generality of the procedure), to be of constant amplitude. Outflow conditions are considered homogeneous on each arc and constant for the entire duration of an interval. By adopting an opportunely short amplitude of the time interval, results are independent of the order in which packets are moved. Once the outflow characteristics on arcs for an interval are known, it is possible to track the movements of vehicles on each arc depending on the assumptions of the type of arc and movement rules defined below. The transport network is represented by means of graph  $G(N, A)$ , where  $N$  is the set of nodes and  $A$  the set of arcs.  $L_a$  is the arc length and  $X_a^S$  the abscissa of a section  $S$  of arc  $a$  identifying the change in movement rule within the arc. The part of the arc in the range  $[0, X_a^S]$  is defined as the *running segment* and the remaining part (range  $[X_a^S, L_a]$ ) as the *queuing segment*. The position of section  $S$  depends on the outflow characteristics of the arc and, in the dynamic assignment model considered, it is evaluated at the beginning of each time interval. Moreover,  $k_a^{max}$  is the maximum density on the arc,  $C_a$  is the arc width and  $Q_a$  is the capacity of the final section of the arc.

A *packet*  $P \equiv \{\eta, od, u\}$ , is characterized by three elements: a departure time  $\eta$ , an origin/destination pair  $od$  and a vehicle class  $u$ . The elements in the packet belong to the same class  $u$ , depart at the same departure time  $\eta$  and move on the same origin/destination pair  $od$ . For each class  $u$  some parameters depending on the characteristics of the vehicles can be defined (speed, occupancy, equivalence, filling, grouping). During the movement, a packet is characterized by a time interval  $t$ ,

a position  $x$  on arc  $a$  belonging to path connecting pair  $od$ , and a speed  $v_a^t$  on the running segment. The simulation is conducted for discrete time intervals  $t$  (assumed at constant amplitude  $\delta$ ), indicating with  $\tau$  the current time ( $\tau \in [0, \delta]$ ). From these considerations, the movement of a packet depends on its position on the section of the arc (running or queuing) and on the fact that, during the movement, it may pass from the running segment to the queuing segment [4]. Moreover, arc capacities, queues, spillback phenomena and overtaking between vehicles of different class are explicitly taken into consideration.

To simulate the movement of a packet  $P$ , for each pair  $od$  and for each departure time  $\eta$ , a Directed Acyclic Graph (DAG)  $\Gamma_{od\eta}(N^{od} \subset N, A^{od} \subset A)$  is associated to packet  $P$  [6]. The DAG consists of a set of arcs that belongs to the feasible paths connecting the origin/destination pair  $od$  computed at time  $\eta$ . Note that (as discussed in the next sections) with the introduction of re-routing capabilities, the DAG is generally time dependent. This because the origin to be considered is the end node of the arc reached by the packet at the current time. The following considerations concern the case when path choice occurs at departure time  $\eta$  and can straightforwardly be generalized to consider re-routing.

Being  $w_a$  the weight of the arc  $a \in A^{od}$ , two cases can be considered for the evaluation of  $w_a$ :

1.  $\Gamma_{od\eta}$  is generated by using an implicit algorithm (i.e. Dial's STOCH) to compute the path set:  $w_a$  can be set equal to the Dial weight;
2.  $\Gamma_{od\eta}$  is generated by using an explicit algorithm to compute the path set:  $w_a$  is provided by the sum of the probabilities of paths crossing arc  $a$  (note that in this case it is necessary to compute path choice probabilities, i.e. by means of a Logit model).

In our experiment the path set is evaluated explicitly. The probability  $\pi_a^\Gamma$  of choosing arc  $a$  belonging to  $\Gamma_{od\eta}$ , given the travel origin  $o$ , the destination  $d$  and the departure time  $\eta$ , can be then defined as:

$$\Pr(a) = \pi_a^\Gamma = \frac{w_a}{\sum_{l \in A_a^{od}} w_l} \quad a \in A^{od} \tag{1}$$

where  $A_a^{od}$  is the set of the arc outgoing from  $a$ .

In order to better simulate path choice during an evacuation, the following capabilities have been added to the mesoscopic model: an arc risk to consider the impacts of the event on the transportation network and a re-routing procedure to choose en-route the next arc of the DAG to be covered.

### 2.1 Arc Risk Function

The risk that can be associated to each arc of the network is evaluated introducing it in the formulation of the arc cost functions [3]. In general the risk probability  $r_R^E(\tau)$  can be associated with the nature of the event ( $E$ ), time ( $\tau$ ) and a geographic position ( $R$ ). In the example reported in this paper, the specific case of atmospheric diffusion of a pollutant is considered, outlining some characteristics of the process. In a urban area, for example, a vehicle transporting dangerous goods having an accident can cause an emergency condition. The effects of the hazard [13] depend on the type of the substance, on weather conditions (i.e. presence of wind, pressure), and on the type of dispersion; they can be evaluated by taking into consideration the concentration of the pollutant at a point in time. In the literature, two main approaches are considered: Eulerian and Lagrangian. Focusing on the Eulerian approach, the propagation problem is solvable under some assumptions (i.e. use of mean value for some variables), considering if the substance is released instantly (puff-based solution) or continuously (plume-based solution). While in the first case the time is considered explicitly, in the latter the solution is not time-dependent. A smoke cloud, moving across an urban area under the influence of the wind, covers an area  $\Omega$  at a time  $\tau$ , uncovering it at a time  $\tau' > \tau$ . In this situation, it is possible to evaluate the substance concentration at each point of the area in time, deriving a measure of the risk associated to the area. Risk  $r_R(\tau)$  at a point  $R$  is defined as the probability that the substance concentration at  $R$  exceeds a threshold value. Being  $R(x, y, z)$  a point in the space,  $c(R, \tau)$  the concentration at point  $R$  at time  $\tau$ ,  $c_{cr}$  a critical value for the concentration, then the risk probability  $r_R(\tau)$  at point  $R$  is evaluable as:

$$r_R(\tau) = \Pr(c > c_{cr}) = \int_{c_{cr}}^{\infty} F(\zeta) d\zeta, \tag{2}$$

where  $F(\zeta)$  is the solution of the Eulerian function.

The risk definition can be extended from a point to an area, defining the risk probability  $r_\Omega(\tau)$  of an area  $\Omega$  at time  $\tau$  as:

$$r_\Omega(\tau) = \max\{r_R(\tau), R \in \Omega\}, \tag{2a}$$

Finally, the risk probability for an arc  $a$  at time  $\tau$  is linked with the area  $\Omega$  containing the arc:

$$r_a(\tau) = r_\Omega(\tau), \quad \forall a \in \Omega \tag{2b}$$

As defined above, the risk probability  $r_a(\tau)$  ( $\in [0, 1]$ ) of the arc  $a$  at time  $\tau$  depend on the vulnerability of the link respect to a hazardous phenomenon  $E$ .

Considering a path  $p$  between an origin  $o$  and a destination  $d$  and assuming the independence among the arcs (that is, what happens on an arc does not depend on

what happens on the others), the risk associated to the path can be expressed as the product of arc risk:

$$r_p(\tau) = \prod_{i \in p} r_i(\tau), \quad (2c)$$

where  $r_i$  is the risk probability on arc  $i$  belonging to the path  $p$ .

With these considerations, it is possible to modify, for each arc, travel time (cost) computed at time  $\tau$ , introducing a weight dependent on the level of risk associated to the arc at time  $\tau$ . At time  $\tau$ , travel time of arc  $a$  can be evaluated as:

$$TT_a(\tau) = x_a^S(\tau)/v_a + ((L_a - x_a^S(\tau)) \cdot k_{\max}^a)/Q_a(\tau), \quad (3)$$

where the first term refers to the running segment and the second to the queuing one. Once travel time is known, the weighted travel time can be written as follows:

$$TW_a(\tau) = TT_a(\tau) \cdot \{1 + \alpha[\ln(1/(1 - r_a(\tau)))]^\beta\}, \quad (4)$$

where  $TW_a(\tau)$  is the weighted time associated to arc  $a$  at time  $\tau$ ;  $\alpha, \beta$  are parameters. The weighted travel time is used to compute the DAG associated to packet  $P$  as defined above, for each origin-destination pair  $od$  and departure time  $\eta$ .

## 2.2 Re-routing

Path choice is a crucial aspect during an evacuation, in order to leave from the affected area. The paths can be designed ex-ante [11], but if the event influence the road network (i.e. some arcs are unavailable) the en-route choice provides alternative paths. The advantage of re-routing is the adaptability to the changed conditions of the network: in fact, re-routing option deals with the possibility of changing the path followed en-route. The set of feasible paths can be reconsidered at each node, and the representation by means of a DAG allows considering all those arcs, belonging to feasible paths, whose origin is the current node. Figure 1 summarizes the re-routing procedure implemented.

Once a packet  $P$  reaches the end node  $j$  of an arc  $a$  the next arc  $a^+$  to be covered must be chosen in order to continue the travel. Arc  $a^+$  belongs to one of the paths reaching the destination. Two cases arise: paths are those connecting the origin-destination pair  $od$  established at departure time  $\eta$  or path set can be modified en route (*re-routing*). In the first case, they are represented by the DAG  $\Gamma_{od\eta}$  associated to packet  $P$  at departure time and the next arc  $a^+$  can be chosen immediately. In the second one, the new paths have to be evaluated (*path search*) starting from the *end node*  $j$  of the arc  $a$  to destination  $d$  at current time  $\zeta$  (when packet  $P$  moves from  $j$ ). The resulting DAG is  $\Gamma_{jd\zeta}$ , and then the next arc  $a^+$  can be chosen.

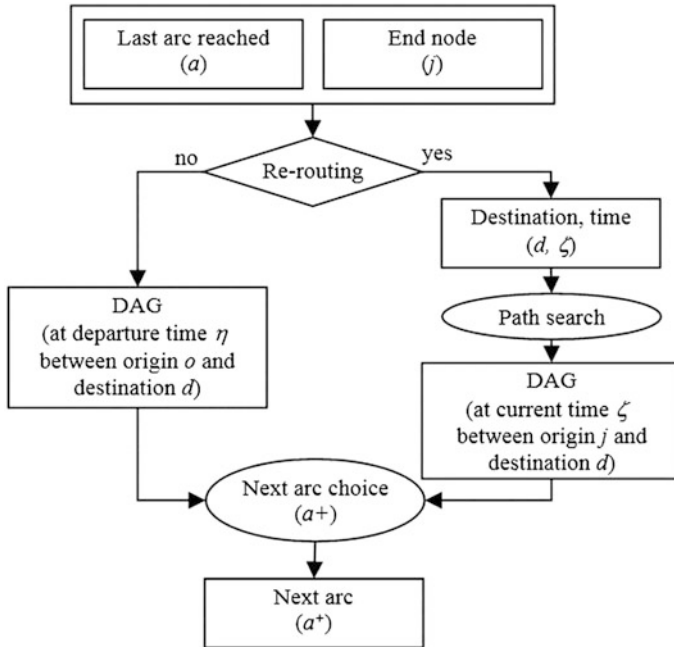
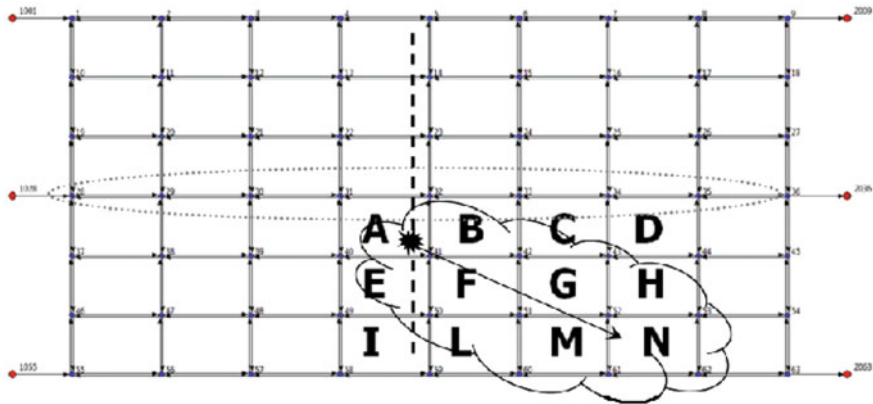


Fig. 1 Procedure adopted to simulate re-routing option

### 3 Application

To give an example of the applicability of the proposed model, an application to a test network was carried out in order to simulate the evolution of impacts on path choice due to a hazardous event whose extents vary in time. In this application, the main characteristics of the trial network (such as length, width, number of lanes, free flow speed) are established randomly and the derived ones (such as capacity) are evaluated consequently. Similarly, trial values for demand and risk probability have been defined. Figure 2 shows the test network, made up of 66 nodes and 232 arcs.

Each arc is characterized by length, width, number of lanes, free flow speed and capacity. In the test network, twelve areas (named from A to N) are selected as involved in the hazardous event. A simulation was conducted with a time interval of 300 s. Demand was generated for the first 27 intervals. To better test the proposed model, origin-destination pairs were chosen in order to generate paths crossing the interested area. The scenario hypothesis consists of a hazardous event occurring at interval  $h = 6$  in the area labelled A (Fig. 2) whose effects (a smoke plume) propagate in the neighbouring areas and make it unsafe to cross roads in the surrounding areas impacted by the evolution of the event. Considering that the risk can be evaluated point to point in the space following Eq. (2), it is possible to



**Fig. 2** Test network and areas involved in hazardous event’s evolution

associate a risk value at each area. The propagation of the effects start from area A at interval  $h = 6$ , risk probabilities are updated at time intervals 6, 9, 15 and 18. At interval  $h = 24$  the effects cease and risk probability is null for all the areas. The values of risk probability are assigned considering Eq. (2b).

To streamline the discussion on obtained results, arc densities are reported focusing on the screen-line highlighted by a dotted line in Fig. 2 and corresponding, from top to bottom, to arcs 4–5, 13–14, 22, 23, 31, 32, 40, 41, 49, 50, 58, 59; the evolution in time of densities for these arcs is shown in Fig. 3. During intervals from 1 to 6, travellers move on the network following those paths defined at the leaving interval. At the beginning of interval 6 the hazardous event happens in area A and, starting from interval 9, a change in the arc densities can be observed since users tend to choose paths that move alongside the affected area. Focusing on destination 2036, it can be observed that some paths reaching it may use arcs 31–40, 40, 41, 31, 32, 32–41 until interval 7. Starting from this interval, until interval 25, other arcs are used. As an example, considering the arcs 31, 32, there is a density of about 20 user/km per lane until interval 6. At interval 7 paths do not pass through this arc but use arc 22, 23. Density decreases on arc 31, 32 and increases on arc 22, 23. Starting from interval 10 no one passes on arc 31, 32 (its density is equal to zero) and density on arc 22, 23 is about 15 user/km per lane. As time passes, density on arc 22, 23 increases up to about 34 user/km per lane (other paths use this arc). At interval 24 the effects on the system expire and the same arcs considered at the beginning of the simulation are used. At interval 33 all the users crossed the screen-line.

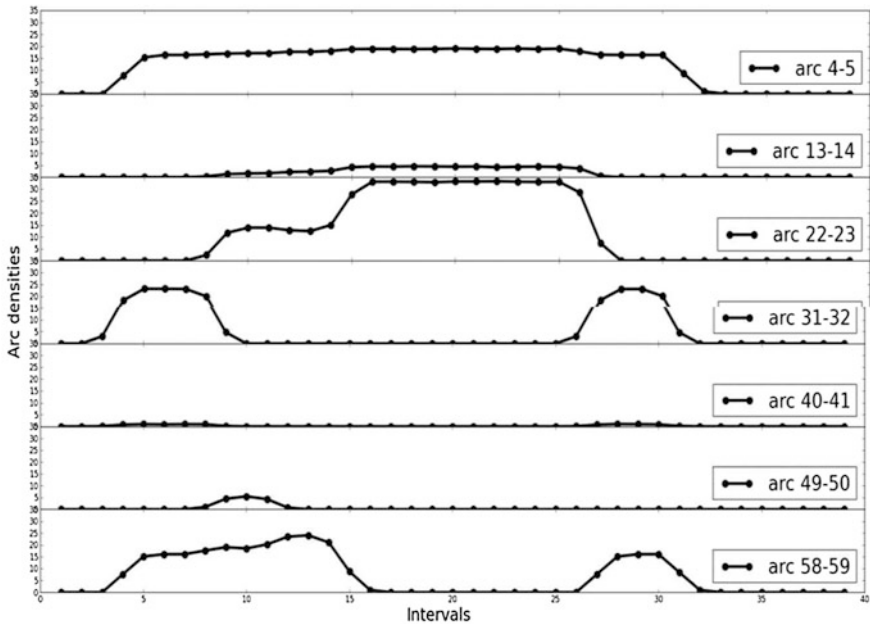


Fig. 3 Evolution of arc densities for each interval

## 4 Conclusions

In this paper, a mesoscopic dynamic network loading model able to simulate travellers' path choice during an evacuation was formulated and tested on a trial network. To achieve the goal, some extensions were introduced to the mesoscopic network loading approach. Path choice was implicitly modelled, to evaluate the path/arc probability and assign the flow simultaneously. The possibility of modelling en-route path choice was explored, introducing an explicit management of re-routing capabilities. A risk factor was also introduced, within arc cost function, to take into account variations in the network conditions. Concerning its application, a scenario hypothesis, consisting of a major event occurred in a time interval within a part of the network (area), was built. The event effects propagate in the neighbouring areas and make it unsafe to cross roads surrounding the area impacted by the evolution of the event. The capability of the proposed model to handle re-routing can be exploited to simulate users' response to external sources of information both on-board and external. The proposed arc cost function can be used to take risk into account even within consolidated simulation tools without making major changes from the modelling point of view.

## References

1. Campos, V., Bandeira, R., Bandeira, A.: A method for evacuation route planning in disaster situations. *Procedia—Soc. Behav. Sci.* **54**, 503–512 (2010)
2. Chilà, G., Musolino, G., Polimeni, A., Rindone, C., Russo, F., Vitetta, A.: Transport models and intelligent transportation system to support urban evacuation planning process. *IET Intel. Transport. Syst.* **10**(4), 279–286 (2016)
3. Di Gangi, M.: Evaluation of reliable path in risk areas. *WIT Trans. Ecol. Environ.* **91**, 371–377 (2006). doi:[10.2495/RISK060351](https://doi.org/10.2495/RISK060351)
4. Di Gangi, M.: Modeling evacuation of a transport system: application of a multi-modal mesoscopic dynamic traffic assignment model. *Trans. Intel. Transport. Syst. IEEE* **12**(4), 1157–1166 (2011)
5. Di Gangi, M., Cantarella, G.E., Di Pace, R., Memoli, S.: Network traffic control based on a mesoscopic dynamic flow model. *Transp. Res. Part C: Emerg. Technol.* **66**, 3–26 (2016)
6. Di Gangi, M., Polimeni, A.: A model to simulate multimodality in a mesoscopic dynamic network loading framework. *J. Adv. Transport.* vol. 2017 (2017). doi:[10.1155/2017/8436821](https://doi.org/10.1155/2017/8436821). Article ID 8436821
7. Fosgerau, M., Frejinger, E., Karlstrom A.: A link based network route choice model with unrestricted choice set. *Trans. Res.* **56**, 70–80 (2009)
8. Hsu, Y.T., Peeta, S.: Behavior-consistent information-based network traffic control for evacuation operations. *Trans. Res.* **48**, 339–359 (2014)
9. Pel, A.J., Hoogendoorn, S.P., Bliemer, M.C.J.: Evacuation modeling including traveler information and compliance behaviour. *Procedia Eng.* **3**, 101–111 (2010)
10. Polimeni, A., Vitetta, A.: Joint network and route optimization in road evacuation. *WIT Trans. Ecol. Environ.* **155**, 1053–1065 (2011)
11. Polimeni, A.: The role of optimization models for rescue vehicles routes in evacuation. *WIT Trans. Inf. Comm. Technol.* **44**, 477–489 (2012)
12. Stepanov, A., MacGregor, Smith J.: Multi-objective evacuation routing in transportation networks. *Eur. J. Oper. Res.* **198**(2), 435–446 (2009)
13. TNO methods for the calculation of physical effects, vol 3rd edn. [www.bib.uv.nl/fileadmin/fdocs/PGS2-1997.pdf](http://www.bib.uv.nl/fileadmin/fdocs/PGS2-1997.pdf) (2005)
14. Wu, H.C., Lindell, M.K., Prater, C.S.: Logistics of hurricane evacuation in Hurricanes Katrina and Rita. *Transp. Res. Part F* **15**, 445–461 (2012)
15. Xu, H., Lam, W.H.K., Zhou, J.: Modelling road users behavioural change over time in stochastic road networks with guidance information. *Transp. B, Trans. Dynam.* **2**(1), 20–39 (2014)



# Last-Mile Deliveries by Using Drones and Classical Vehicles

Luigi Di Puglia Pugliese and Francesca Guerriero

**Abstract** We address the problem of managing a drone-based delivery process. We consider the specific situation of a delivery company, that uses a set of trucks equipped with a given number of drones. In particular, items of a limited weight and size could be delivered by using drones. A vehicle, during its trip, can launch a drone when serving a customer, the drone performs a delivery for exactly one customer and returns to the vehicle, possibly at a different customer location. Each drone can be launched several times during the vehicle's route. It is imposed a limit on the maximum distance that each drone can travel and synchronization requirements between vehicle and drone should be ensured. In particular, it is assumed that a vehicle waits for a drone for a maximum period of time. The aim is to serve all customers within their time window. The problem is modeled as a variant of the vehicle routing problem with time windows. The aim of this work is to analyze the delivery process with drones, by taking into account the total transportation cost and highlighting strategic issues, related to the use of drones. The numerical results, collected on instances generated to be very close to reality, show that the use of drones is not economically convenient in the classical terms. However, when considering negative externalities related to the use of classical vehicles and quality of service requirements, the benefit of using drones becomes relevant.

**Keywords** Vehicle routing problem · Time windows · Drone · Last-mile delivery

## 1 Introduction

The scientific literature gave great attention to the distribution problems, encountered in the last-mile delivery process. The distribution problem of items in an urban

---

L. Di Puglia Pugliese (✉) · F. Guerriero (✉)  
DIMEG - University of Calabria, Rende, Italy  
e-mail: luigi.dipugliapugliese@unical.it

F. Guerriero  
e-mail: francesca.guerriero@unical.it

area is referred to as vehicle routing problem (*VRP*) when the trucks have a limited capacity. On the other hand, the scientific literature refers to the travelling salesman problem (*TSP*) when only a vehicle with an unlimited capacity is used. Depending on the specific scenario under consideration, several operational constraints have to be taken into account, such as, time windows, precedence, packing. For more detail on the *VRP* and *TSP* and their variants, the reader is referred to [1, 2].

In the last two decades, we have seen a surge in direct-to-consumer deliveries, due to the continued growth of e-commerce, the rapid urbanization and the heightened customer expectations. These trends have introduced new optimization challenges in last-mile delivery process that became more complex to manage.

On the other hand, in the last years, drone technology has seen important advancements and several companies interested in package delivery, like Amazon [3, 4], Federal Express [5], DHL [6] have begun investigating the possibility of using drones for their distribution service.

The scientific literature has started to study the last-mile drone-based delivery process [7–9], by formulating and solving some variants of the *TSP*. In [10], the authors consider a fleet of vehicles focusing their attention on the duration of the process and analyzing the theoretical worst-case, under several scenarios.

In this paper, we formulate the truck-drone delivery problem as a variant of the vehicle routing problem with time windows (*VRPTW*) where each vehicle is equipped with drones (*VDRPTW*). The aim is to gain quantitative insights, in order to show potential benefits and disadvantages in using drones in the distribution process. We consider the minimization of the total transportation cost and we show numerical results, that highlight the advantages/drawbacks in the use of drones in the last-mile delivery service.

The reminder of the paper is organized as follows. In Sect. 2, we describe the problem under study, providing the main assumptions on the way the drones are used. Then, we present its mathematical formulation. Section 3 reports the computational results collected on instances generated to be very close to real-life applications. Section 4 concludes the paper.

## 2 Problem Definition

In order to describe the *VDRPTW*, it is useful to introduce the assumptions made regarding the behavior of the drones and the cooperation between truck and drone.

- A drone-delivery can serve at most one customer at time.
- A drone is able to perform several drone-deliveries.
- The set-up time for preparing the drone for a new drone-delivery is negligible.
- The drone can wait in the ground of a customer for a given maximum time.
- After a delivery, the drone must return to the own truck located at some customer.
- During a drone-delivery, the truck performs its route.
- A truck cannot visit a customer served by a drone to pick-up the drone.

We formulate the *VDRPTW* over a complete graph  $G = (V, A)$ , where  $V$  is the set of nodes and  $A$  is the set of arcs. The set  $V$  contains  $n$  nodes associated with the customers, named  $N$ , and two extra nodes 0 and  $n + 1$  representing the depot. We refer to  $V_L = \{0, 1, \dots, n\}$  as the set containing all nodes from which the drones can start their deliveries, and to  $V_R = \{1, 2, \dots, n + 1\}$  as the subset of nodes  $V$  where the drones return to the truck after performing their delivery. In addition, we denote with  $\bar{N} \subseteq N$  the set containing the nodes associated with the customer that can be served by a drone.

We define the parameters  $d_{ij}$  and  $\bar{d}_{ij}$  to indicate the distance to reach node  $j$  from node  $i$  considering the truck and the drone, respectively. We assume  $\bar{d}_{ij} \leq d_{ij}$ , because the drone does not necessarily follow the road. The time spent to traverse the arc  $(i, j)$  is defined for the truck and the drone as  $t_{ij} = d_{ij}/v$  and  $\bar{t}_{ij} = \bar{d}_{ij}/\bar{v}$ , respectively, where  $v$  and  $\bar{v}$  are the average speeds of the truck and the drone.

A demand  $q_i$  is associated with each node  $i \in N$ . In addition, we assume a service time for each customer served by the truck, named  $s_i$ , and a service time  $\bar{s}_i$  if the customer  $i$  is served by the drone. A time window is associated with each customer  $i \in N$ . We indicate with  $l_i$  and  $u_i$  the opening and the closing time for serving customer  $i$ .

We assume the availability of a limited number of trucks, belonging to the set  $K$  and a limited number of preassigned drones to each truck, named  $D$ . Each drone has a limited autonomy in term of maximum distance  $E$  that it can travel. In addition, a drone is able to remain at a customer location for a maximum time  $T$  before and after the delivery takes place.

A drone-delivery is characterized by the triple  $(i, w, j)$  where  $i \in V_L$  is the node where the drone starts its delivery,  $w \in \bar{N}$  is the node associated with the customer served by the drone, and  $j \in V_R$  is the node where the drone returns to the truck, with  $i \neq w \neq j$ .

The aim is to analyze the last-mile delivery service with the aid of drones under an economic point of view. Thus, the objective function (to be minimized) represents the total cost deriving from the use of the trucks ( $C^1$ ) and the drones ( $C^2$ ). It is worth observing that in the proposed model a function of the cost per unit of distance travelled is used and thus several metrics can be adopted. In particular, it is possible to consider not only monetary values but also pollution, noise, congestion and others negative externalities.

The variables used to represent mathematically the *VDRPTW* are reported in the following:

- $x_{ij}^k, \forall i, j \in V, k \in K$  Binary variables indicating whether arc  $(i, j)$  belongs to the route of truck  $k$ .
- $y_{iwj}^k, \forall i, w, j \in V, k \in K$  Binary variables indicating whether drone-delivery  $(i, w, j)$  is performed by a drone associated with truck  $k$ .
- $p_{ij}^k, \forall i, j \in V, k \in K$  Binary variables indicating whether node  $i$  is served before but not necessarily consecutive to node  $j$  in the route of truck  $k$ .

- $u_i^k, \forall i \in V, k \in K$  Integer variables indicating the position of node  $i$  in the route of truck  $k$ .
- $z_{aiwje}^k, \forall a, i, w, j, e \in V, k \in K$  Binary variables indicating whether the drone-delivery  $(i, w, j)$  is performed by a drone associated with vehicle  $k$  and node  $a$  precedes node  $i$  and node  $e$  follows immediately node  $j$ .
- $a_i^k, \forall i \in V, k \in K$  Continuous variables indicating the instant time in which customer  $i$  is served by either the truck  $k$  or a drone associated with truck  $k$ .

On the basis of the notation introduced above the *VDRPTW* can be mathematically represented as follows.

$$\min C^1 \sum_{i \in V_L} \sum_{j \in V_R} \sum_{k \in K} d_{ij} x_{ij}^k + C^2 \sum_{i \in V_L} \sum_{j \in V_R} \sum_{w \in \bar{N}} \sum_{k \in K} (\bar{d}_{iw} + \bar{d}_{wj}) y_{iwj}^k \quad (1)$$

$$\sum_{j \in N} x_{0j}^k - \sum_{i \in N} x_{i,n+1}^k = 0, \forall k \in K \quad (2)$$

$$\sum_{i \in V_L} x_{ih}^k - \sum_{j \in V_R} x_{hj}^k = 0, \forall h \in N, k \in K \quad (3)$$

$$\sum_{j \in N} x_{0j}^k \leq 1, \forall k \in K \quad (4)$$

$$\sum_{i \in V_L} \sum_{k \in K} x_{ij}^k + \sum_{l \in V_L} \sum_{m \in V_R} \sum_{k \in K} y_{ljm}^k = 1, \forall j \in N \quad (5)$$

$$2y_{iwj}^k \leq \sum_{h \in V_L} x_{hi}^k + \sum_{l \in N} x_{lj}^k, \forall k \in K, i \in N, w \in \bar{N}, j \in V_R \quad (6)$$

$$y_{0wj}^k \leq \sum_{h \in V_L} x_{hj}^k, \forall w \in \bar{N}, j \in V_R, k \in K \quad (7)$$

$$u_i^k - u_j^k \leq (n+1)(1 - x_{ij}^k), \forall k \in K, i \in N, j \in V_R \quad (8)$$

$$u_j^k - u_i^k \geq 1 - (n+1)(1 - y_{iwj}^k), \forall k \in K, i \in N, w \in \bar{N}, j \in V_R \quad (9)$$

$$u_i^k - u_j^k \geq 1 - (n+1)p_{ij}^k, \forall k \in K, i \in N, j \in V_R \quad (10)$$

$$u_i^k - u_j^k \leq -1 + (n+1)(1 - p_{ij}^k), \forall k \in K, i \in N, j \in V_R \quad (11)$$

$$3z_{aiwje}^k \leq y_{awe}^k + p_{ai}^k + p_{je}^k, \forall k \in K, a \in V_L, i \in N, w \in \bar{N}, j \in N, e \in V_R \quad (12)$$

$$2 + z_{aiwje}^k \geq y_{awe}^k + p_{ai}^k + p_{je}^k, \forall k \in K, a \in V_L, i \in N, w \in \bar{N}, j \in N, e \in V_R \quad (13)$$

$$M_{ij}(x_{ij}^k - 1) + a_i^k + s_i + t_{ij} \leq a_j^k, \forall k \in K, i \in V_L, j \in V_R \quad (14)$$

$$M_{iw}(y_{iwj}^k - 1) + \bar{t}_{iw} + a_i^k \leq a_w^k, \forall k \in K, i \in V_L, w \in \bar{N}, j \in V_R \quad (15)$$

$$M_{wj}(y_{iwj}^k - 1) + a_w^k + \bar{s}_w + \bar{t}_{wj} \leq a_j^k, \forall k \in K, i \in V_L, w \in \bar{N}, j \in V_R \quad (16)$$

$$M_{iw}(y_{iwj}^k - 1) + a_w^k - \bar{t}_{iw} - a_i^k \leq T, \forall k \in K, i \in V_L, w \in \bar{N}, j \in V_R \quad (17)$$

$$M_{wj}(y_{ij}^k - 1) + a_j^k - \bar{t}_{wj} - a_w^k - \bar{s}_w \leq T, \forall k \in K, i \in V_L, w \in \bar{N}, j \in V_R \quad (18)$$

$$(\bar{d}_{iw} + \bar{d}_{wj})y_{ij}^k \leq E, \forall k \in K, i \in V_L, j \in V_R, w \in \bar{N} \quad (19)$$

$$\sum_{i \in V_L} q_i \sum_{j \in V_R} x_{ij}^k + \sum_{w \in \bar{N}} q_w \sum_{i \in V_L} \sum_{j \in V_R} y_{ij}^k \leq C, \forall k \in K \quad (20)$$

$$M_{ij}(x_{ij}^k - 1) + \sum_{l \in \bar{N}} \sum_{m \in V_R} y_{ilm}^k + \sum_{p \in V_L} \sum_{q \in \bar{N}} y_{pqj}^k + \sum_{w \in \bar{N}} y_{iwj}^k + \sum_{a \in V_L} \sum_{w \in \bar{N}} \sum_{e \in V_R} z_{aiwje}^k \leq D,$$

$$\forall k \in K, i \in V_L, j \in V_R \quad (21)$$

$$l_j \leq a_j^k \leq u_j, \forall k \in K, j \in N \quad (22)$$

The objective function (1) minimizes the total transportation cost. Constraints (2)–(4) define the route for each truck  $k \in K$ . Constraints (5) impose that all customers have to be served. Constraints (6) and (7) define the variables  $y$ . In particular, constraints (6) impose that a drone-delivery  $(i, w, j)$  for truck  $k \in K$  is performed only if nodes  $i$  and  $j$  belong to the route of truck  $k \in K$ . Constraints (7) manage the situation where the drone starts its delivery from the depot. Constraints (8) and (9) define the order in which the customers are served by the route performed by truck  $k$ . Constraints (10) and (11) define the variables  $p$ . In particular, if  $u_j^k > u_i^k$ , then constraints (10) ensure that  $p_{ij}^k = 1$ . On the other hand, if  $u_j^k < u_i^k$ , then both constraints (10) and (11) are satisfied for  $p_{ij}^k = 0$ . Constraints (12) and (13) specify variables  $z$ . Constraints (14)–(18) assign the starting service time at each customer. In particular, the constraints (14) and (15) define the starting service time at customer served by the truck and the drone, respectively. Constraints (14) and (16) guarantee the synchronization between the truck and the drone at customer  $j$  when the drone-delivery  $(i, w, j)$  is performed. Constraints (17) and (18) impose a maximum waiting time of  $T$  for the drone-delivery  $(i, w, j)$ . Constraints (19) select feasible drone-delivery with respect to the total distance travelled. Constraints (20) represent the capacity constraints for the truck. Constraints (21) guarantee that at most  $D$  drone-delivery are performed simultaneously. Constraints (22) guarantee the satisfaction of the time windows requirement.

### 3 Computational Results

Model (1)–(22) is implemented in Java language and solved by using the CPLEX 12.5.1 library. The aim of the computational phase is to assess the behavior of the proposed model in terms of solution quality. In particular, a sensitivity analysis on the solution obtained when some fundamental parameters are modified is carried out. To this aim, several scenarios, very close to the real-life and satisfying the assumptions reported in the previous section, are considered.

The analysis is conducted from the efficiency point of view related to total transportation cost. A comparison among the solutions obtained by solving the classical *VRPTW* and those of the proposed *VDRPTW* is carried out.

The main aim of this section is to provide some quantitative solutions that allow to highlight strategical issues and provide insights between benefits and factor of risk, in the use of drone in the last-mile delivery service. The tests are carried out on an Intel Core i7-4720HQ 2.60 GHz 8.00 GB RAM, under Microsoft Windows 10 operating system.

### 3.1 Instances

We have generated instances very close to real-life by considering the parameters listed below.

- $F$  Dimension of the field. It is assumed that the customers are located in an area of  $F \times F$  miles<sup>2</sup>. The computational experiments are carried out by considering  $F = \{10, 20\}$ .
- $|N|$  Number of customer. In the experiments, two values for  $|N|$ , that is, 5 and 10 are used.
- $|\bar{N}|$  Number of customers with feasible demand for a drone-delivery. In our experimental phase we set  $|\bar{N}| = 0.8|N|$ .
- $\alpha$  Percentage of customers in  $\bar{N}$  that can be served by a drone. The parameter  $\alpha$  is chosen from the set  $\{0.50, 0.75, 1.00\}$ . This parameter is used to model the situations in which, even though the customer has a demand feasible with the drone-delivery, the visit of a drone is forbidden for others reasons, such as, ground of the customer location, explicitly request of the customer.
- $v$  Average speed of the trucks (25 mph).
- $\bar{v}$  Average speed of the drones (55 mph).
- $C$  Capacity of the truck ( $C = 50$ ).
- $E$  Maximum distance for a drone-delivery (15 miles).
- $s_i$  Service time at customer  $i$  for the truck (2 min).
- $\bar{s}_i$  Service time at customer  $i$  for the drone (1 min).
- $T$  Maximum waiting time (4 min).

The number of available vehicle, i.e.  $|K|$  is set equal to 2. The width of the time windows  $u_i - l_i$  is chosen equal to 4 h for each customer  $i$ . The location of the customer  $i$ ,  $(X_i, Y_i)$ , is imposed by considering values for  $X_i$  and  $Y_i$  belonging to  $[0, F]$  and chosen by considering a uniform distribution. The demand  $q_i$  is set to 1, for  $i \in \bar{N}$ , whereas  $q_i \in [2, 10]$ , for all customers  $i \in N \setminus \bar{N}$ . The parameters  $d_{ij}$  and  $\bar{d}_{ij}$  are set equal to the Manhattan and Euclidean distance, respectively. In order to evaluate the benefits and the disadvantages in using drones, we drop the constraint related to the maximum number of drones available for each vehicle.

### 3.2 Numerical Results

We present numerical results collected by solving to optimality the proposed model.

Table 1 reports average computational results over all instances with the same number of nodes, parameters  $F$  and  $\alpha$ , respectively. Column costs reports the objective function value, column  $\beta$  shows the maximum arrival time at the depot, evaluated among all vehicles, that is,  $\beta = \max_{k \in K} \{a_{n+1}^k\}$ , column #drn reports the number of drone-delivery, column #vehicles shows the number of vehicles activated, column time reports the execution time, column var % shows the percentage cost increase if the drone-delivery are forbidden (solution of the *VRPTW*).

The parameter  $c$  indicates the ratio between cost  $C^1$  and  $C^2$  ( $c = C^1/C^2$ ). The computational results of Table 1 are collected considering  $C^2 = 1$ . Increasing the value of  $c$  means that more attention is paid to externality costs.

The results indicate that the use of drones is not a viable alternative to the use of classical vehicles under the assumption made and the instances considered in this paper. Indeed, when the transportation cost is the same for the vehicles and the drones (see rows with  $c = 1$ ), the optimal solution of *VDRPTW* does not include drones.

The benefits of using drones increase when the ratio  $c$  increases. Indeed, the average number of drone-delivery is 1.50 and 1.75 for  $c$  equal to 9 and 25, respectively.

The results highlight a clear benefit when drones are included in the solution in term of completion time. Indeed, column  $\beta$  suggests that the duration of the longest route for the *VRPTW* is greater than the longest route obtained when drones are used. In particular,  $\beta$  for the *VRPTW* is 1.10 times higher than the value of  $\beta$  for the *VDRPTW*, for both  $c$  equal to 9 and 25.

This result is coherent with the worst-case analysis made in [10]. However, it is worth observing that the authors in [10] consider the minimization of the longest route and relax several assumptions, made in this paper, such as the maximum distance and the maximum waiting time for the drone, and the time window constraints.

From the results collected we can draw the following considerations: (1) the use of drones in last-mile delivery is strongly affected by the cost; (2) to increase the use of drones the transportation system should be viewed under a green perspective; (3) the drones are necessary in the case a customer cannot be reached by a classical vehicle; and (4) quality of service metrics could increase the benefits in using drones.

## 4 Conclusions

In this paper we investigate the problem of managing a fleet of vehicles equipped with drones. We formulate a mathematical model taking into account several operational constraints related to the technology of drones and to the cooperation between drone and vehicle. In addition, we consider time windows associated with each customer to be served. We analyze the problem investigating the potential of using drone in last-mile delivery process highlighting benefits and drawbacks.

**Table 1** Average computational results comparing solutions of the *VDRPTW* and the *VRPTW*

		<i>VDRPTW</i>						<i>VRPTW</i>					
		Costs	$\beta$	#vehicles	#drn	Time	Var (%)	$\beta$	#vehicles	Time			
c = 1	#nodes	59.87	153.88	1.00	0.00	0.27	0.00	153.88	1.00	0.06			
		62.56	170.32	1.00	0.00	2.16	0.00	170.32	1.00	0.41			
	F	40.79	113.02	1.00	0.00	1.57	0.00	113.02	1.00	0.24			
	$\alpha$	81.63	211.18	1.00	0.00	0.86	0.00	211.18	1.00	0.22			
	0.50	61.21	162.10	1.00	0.00	1.16	0.00	162.10	1.00	0.18			
	0.75	61.21	162.10	1.00	0.00	1.37	0.00	162.10	1.00	0.25			
	1.00	61.21	162.10	1.00	0.00	1.56	0.00	162.10	1.00	0.22			
AVG		<b>61.21</b>	<b>162.10</b>	<b>1.00</b>	<b>0.00</b>	<b>1.36</b>	<b>0.00</b>	<b>162.10</b>	<b>1.00</b>	<b>0.22</b>			
c = 9	#nodes	509.93	144.31	1.00	1.00	0.24	5.65	153.88	1.00	0.06			
		524.64	152.60	1.13	1.75	12.73	8.59	170.32	1.00	0.41			
	F	338.25	99.12	1.00	1.75	11.48	8.73	113.02	1.00	0.24			
	$\alpha$	696.32	197.78	1.13	1.00	1.49	5.51	211.18	1.00	0.22			
	0.50	522.28	150.52	1.00	1.00	1.61	5.48	162.10	1.00	0.18			
	0.75	513.41	147.12	1.00	1.25	2.96	8.40	162.10	1.00	0.25			
	1.00	511.17	145.65	1.25	2.25	20.52	9.13	162.10	1.00	0.22			
AVG		<b>515.62</b>	<b>147.77</b>	<b>1.07</b>	<b>1.50</b>	<b>8.36</b>	<b>7.67</b>	<b>162.10</b>	<b>1.00</b>	<b>0.22</b>			
c = 25	#nodes	1398.87	144.31	1.00	1.00	0.27	6.99	153.88	1.00	0.06			
		1425.97	151.81	1.00	2.13	20.33	12.38	170.32	1.00	0.41			
	F	910.98	98.33	1.00	2.13	19.18	12.73	113.02	1.00	0.24			
	$\alpha$	1913.86	197.78	1.00	1.00	1.42	6.64	211.18	1.00	0.22			
	0.50	1437.84	150.52	1.00	1.00	2.24	6.43	162.10	1.00	0.18			
	0.75	1400.93	145.54	1.00	2.00	3.84	11.14	162.10	1.00	0.25			
	1.00	1373.09	145.65	1.00	2.25	34.26	14.74	162.10	1.00	0.22			
AVG		<b>1403.95</b>	<b>147.24</b>	<b>1.00</b>	<b>1.75</b>	<b>13.45</b>	<b>10.77</b>	<b>162.10</b>	<b>1.00</b>	<b>0.22</b>			



## References

1. Toth, P., Vigo, D.: Vehicle routing: problems, methods, and applications. MOS-SIAM Series on Optimization 18, 2nd edn. SIAM, Philadelphia (2014)
2. Applegate, D.L., Bixby, R.E., Chvatal, V., Cook, W.J.: The Traveling Salesman Problem: A Computational Study. Princeton University Press, New Jersey (2006)
3. Popper, B.: Drones could make Amazons dream of free delivery profitable (2015). <http://www.theverge.com/2015/6/3/8719659/amazon-prime-air-drone-delivery-profit-freeshipping-small-items>
4. BBC News: Amazon testing drones for deliveries (2013). <http://www.bbc.com/news/technology-25180906>
5. Bermingham, F.: Fedex researching drone delivery but not for widespread use (2014). <http://www.ibtimes.co.uk/fedex-researching-drone-delivery-not-widespread-use-1471063>
6. DHL Press Release: DHL parcelcopter launches initial operations for research purposes (2014). [http://www.dhl.com/en/press/releases/releases\\_2014/group/dhl\\_parcelcopter\\_launches\\_initial\\_operations\\_for\\_research\\_purposes.html](http://www.dhl.com/en/press/releases/releases_2014/group/dhl_parcelcopter_launches_initial_operations_for_research_purposes.html)
7. Gambella, C., Lodi, A., Vigo, D.: Exact Methods for Solving the Carrier-Vehicle Traveling Salesman Problem (CVTSP). Presented at VeRoLog Meeting, Vienna (2015)
8. Murray, C.C., Chu, A.G.: The flying sidekick traveling salesman problem: optimization of drone assisted parcel delivery. *Trans. Res. Part C Emerg. Technol.* **54**, 86–109 (2015)
9. Agatz, N., Bouman, P., Schmidt, M.: Optimization Approaches for the Truck and Drone Delivery Problem. Presented at the INFORMS TSL Workshop, Berlin (2015)
10. Wang, X., Poikonen, S., Golden, B.: The vehicle routing problem with drones: several worst-case results. *Optim. Lett.* (2016). <https://doi.org/10.1007/s11590-016-1035-3>

# A Scenario Planning Approach for Shelter Location and Evacuation Routing

Annunziata Esposito Amideo and Maria Paola Scaparra

**Abstract** Emergency planning operations are one of the key aspects of Disaster Operations Management (DOM) [1]. This work presents a scenario-based location-allocation-routing model to optimize evacuation planning decisions, including where to establish shelter sites and which routes to arrange to reach them, across different network disruption scenarios. The model considers both supported-evacuation and self-evacuation. The objective is to minimize the duration of the supported-evacuation while guaranteeing that the routes of self-evacuees do not exceed a given traveling time threshold. Both shelter location and routing decisions are optimized so as to identify solutions which perform well across different disruption scenarios. A mathematical formulation of this model is provided, which can be solved through a general-purpose solver optimization package for modest size instances. Some computational results are reported.

**Keywords** Disaster management · Evacuation planning · Shelter location

## 1 Introduction

The fast-paced increase at which disastrous events, either natural or man-made [19], have occurred in recent years [13], ranging from the World Trade Center terroristic attack in NY (USA, 2001) to the most recent earthquakes in Amatrice (Italy, 2017), claims for more attention to emergency evacuation planning.

Emergency planning operations are one of the key aspects of Disaster Operations Management (DOM) [1]. They include activities such as selecting potential shelter sites and identifying evacuation routes.

---

A. Esposito Amideo (✉) · M.P. Scaparra  
Kent Business School, University of Kent, Canterbury, Kent, UK  
e-mail: ae306@kent.ac.uk

M.P. Scaparra  
e-mail: m.p.scaparra@kent.ac.uk

When planning for shelter location and vehicle-based evacuation, three main issues should be tackled: (1) where and how many shelters should be opened; (2) how should self-evacuation be addressed in the planning framework? (3) how should supported-evacuation be organized, to assist people belonging to sensitive categories (e.g., disabled, elderly)? It is clear that these three issues are highly interconnected and must be addressed simultaneously. Self-evacuees and supported-evacuees, in fact, must share the same resources (capacitated shelters, evacuation routes, etc.).

In this paper, we present a scenario-based location-allocation-routing model to optimize evacuation planning decisions, including where to open shelters and how to route evacuees to them, across different network disruption scenarios. The scenarios are used to capture the uncertainty characterizing road conditions in the aftermath of a disaster. Although both shelter location and evacuation routing operations belong to the disaster response phase, shelters must often be set up and equipped with personnel and relief supplies when the disaster is still evolving and road conditions are uncertain or subject to changes. It is therefore paramount to identify shelter locations which are easily accessible in different disruption scenarios and guarantee an efficient evacuation in every scenario.

The remainder of this paper is organized as follows. Section 2 provides a brief literature review of shelter location and evacuation routing problems. The mathematical formulation of the proposed model and some computational results are reported in Sects. 3 and 4, respectively. Section 5 offers some conclusive remarks.

## 2 Literature Review

Shelter location problems aim at determining the optimal locations of shelter sites while minimizing the travelling time (or distance) [7] between evacuation zones and shelters. Briefly, a shelter is a facility where people belonging to a community stricken by a disaster can seek different kinds of services (e.g., first-aid treatment, food, etc.). Shelter location problems are usually modelled through a location model, typically the  $p$ -median model or one of its variants [5, 7].

Evacuation routing problems aim at determining the set of optimal routes to be travelled. The most common types of evacuation considered in the literature are: (1) self-evacuation (or car-based evacuation), and (2) supported-evacuation (or bus-based evacuation). As reported in [18], car-based evacuation is normally represented by minimum cost network flow models [6], while bus-based evacuation is represented by vehicle routing models [3, 8, 10, 12].

In an efficient evacuation plan, shelter and evacuation routing decisions should be tackled together. Studies on shelter location and car-based evacuation [2, 7, 14–16] are more common than studies merging shelter location with bus-based evacuation [11]. To the best of our knowledge, the only paper that addresses shelter location, car- and bus-based evacuation into an integrated model is [9].

The aim of our study is to propose an alternative integrated model, which combines the use of a system-optimal approach and a user-optimal approach. Specifically, we assume that the evacuation planner has control over the bus routes. However, car-based evacuees aim at reaching the shelters following the shortest available path. The model in [9], instead, uses only a system-optimal approach, where the planning authority controls the behavior of both self- and supported-evacuees. In addition, our model accounts for road disruption by including different scenarios (in each scenario only a subset of links is available) and identifies solutions that are robust under different disaster conditions (as in [16]).

In this initial integrated model, we do not consider road congestion. However, we assume that the traffic on the roads directly leading to a shelter is eased by a contraflow lane reversal. Contraflow lane reversal is commonly used in emergency situations to increase the number of road lanes exiting the disaster zone [18].

Finally, we assume that bus-based demand is sparse and the number of people in need of supported-evacuation is a small fraction of the total number of evacuees. These imply that buses can collect people from different evacuation areas before going to the shelters and that the number of available buses is sufficient to bring all the evacuees outside the dangerous zone with a single trip for each bus (due to our contraflow reversal assumption, buses cannot go back to the dangerous zone).

### 3 Model Formulation

In this section, we present a mathematical formulation for the Scenario-Indexed Shelter Location and Evacuation Routing (SISLER) problem. The assumptions underpinning the models are as follows.

1. Both self-evacuation and supported-evacuation are considered.
2. Self-evacuation involves people evacuating with their own vehicles towards a shelter (people moving towards other destinations are not considered). We refer to this type of evacuation as evacuation mode (a) or car-based evacuation.
3. Supported-evacuation is arranged by public authorities and relies on buses which are stored and dispatched from a depot. We refer to this type of evacuation as evacuation mode (b) or bus-based evacuation.
4. The area affected by the disaster is divided in different evacuation zones. Both self- and supported-evacuation start at the centroid of each zone.
5. For each zone, the number of self-evacuees and bus-evacuees (evacuation demand) is known.
6. Both shelters and buses have a limited capacity.
7. Split delivery of supported-evacuees is possible (more than one bus may collect people from the same area and bring them to different shelters). However, all self-evacuees from the same zone go to the same shelter. From a practical point of view, in fact, it would be difficult to direct self-evacuees to different shelters.

8. The objective is to minimize the supported-evacuation completion time.
9. Self-evacuees use the shortest available path to reach their assigned shelters and, to guarantee an egalitarian allocation, they must reach shelter sites within a given travel time threshold.
10. Contraflow [4, 17] has been assumed on the network arcs whose destination node is a shelter site, which means that those arcs can be travelled only in one direction towards the shelter.
11. Each bus performs a single trip to collect evacuees from the demand points.
12. Several disruption scenarios are considered, which differ in terms of road link availability.

The SISLER problem can be described as follows. Given a directed network  $G(N, A)$ , where  $N$  is the set of nodes and  $A$  is the set of arcs, let  $N_a$  ( $N_a \subseteq N$ ) and  $N_b$  ( $N_b \subseteq N$ ), indexed by  $i$ , be the sets of zones where evacuation mode (a) and evacuation mode (b) start, respectively;  $N_s$  ( $N_s \subseteq N$ ) the set of potential shelter sites, indexed by  $j$ ;  $D$ , the set of network disruption scenarios, indexed by  $d$ ;  $A_d$  ( $A_d \subseteq A$ ) the set of available arcs under disruption scenario  $d$ ; and  $K$  the set of buses stored and dispatched from a depot node  $o$  ( $o \in N$ ), indexed by  $k$ .

The model parameters are:  $q_i^a$  ( $i \in N_a$ ) and  $q_i^b$  ( $i \in N_b$ ) are the expected number of mode (a) and mode (b) evacuees in zone  $i$ , respectively;  $C_j$  and  $r_j$ , the capacity of a shelter and the amount of resources to set up a shelter at site  $j$ ;  $R$  the total amount of available resources;  $B_k$  the capacity of bus  $k$ ;  $\tau_{lm}^d$  the travel time from node  $l$  to node  $m$  in scenario  $d$  (for bus-based evacuation);  $t_{ij}^d$  the shortest travel time from zone  $i \in N_a$  to site  $j$  in scenario  $d$  (for car-based evacuation);  $T^d$  the travel time threshold for self-evacuees in scenario  $d$ ;  $\alpha$  the tolerance parameter to analyze variations of the threshold  $T^d$ ;  $p_d$  the probability of occurrence of scenario  $d$ .

The model uses the following flow variables: for each bus  $k$  in scenario  $d$ ,  $g_{lm}^{kd}$ ,  $v_i^{kd}$ , and  $w_j^{kd}$ , are, respectively, the number of evacuees who travel from node  $l$  to node  $m$ , start evacuation at node  $i \in N_b$ , and end evacuation at site  $j$ . In addition, it uses the location variables  $y_j$ , equal to 1 if a shelter is opened at site  $j$ , 0 otherwise; the allocation variables  $x_{ij}^d$ , equal to 1 if the evacuees in zone  $i \in N_a$  are assigned to shelter  $j$  in scenario  $d$ , 0 otherwise; and the routing variables  $z_{lm}^{kd}$ , equal to 1 if bus  $k$  travels from node  $l$  to node  $m$  in scenario  $d$ , 0 otherwise. Finally, the model uses the variables  $\gamma_d$  to represent the bus-based evacuation completion time in scenario  $d$ .

The mathematical formulation is:

$$\begin{aligned} \min \quad & \sum_{d \in D} P_d \gamma_d \\ \text{s.t.} \quad & \end{aligned} \tag{1}$$

$$\gamma_d \geq \sum_{(l,m) \in A_d} \tau_{lm}^d z_{lm}^{kd} \quad \forall k \in K, d \in D \tag{2}$$

$$\sum_{j \in N_s} x_{ij}^d = 1 \quad \forall i \in N_a, d \in D \quad (3)$$

$$x_{ij}^d \leq y_j \quad \forall i \in N_a, j \in N_s, d \in D \quad (4)$$

$$\sum_{j \in N_s} t_{ij}^d x_{ij}^d \leq (1 + \alpha) T^d \quad \forall i \in N_a, d \in D \quad (5)$$

$$\sum_{m: (o,m) \in A_d} z_{om}^{kd} \leq 1 \quad \forall k \in K, d \in D \quad (6)$$

$$v_m^{kd} + \sum_{l: (l,m) \in A_d} g_{lm}^{kd} = w_m^{kd} + \sum_{l: (m,l) \in A_d} g_{ml}^{kd} \quad \forall m \in N, k \in K, d \in D \quad (7)$$

$$\sum_{l: (l,m) \in A_d} z_{lm}^{kd} - \sum_{l: (m,l) \in A_d} z_{ml}^{kd} = 0 \quad \forall m \in N(o \cup N_s), k \in K, d \in D \quad (8)$$

$$g_{lm}^{kd} \leq B_k z_{lm}^{kd} \quad \forall (l,m) \in A_d, k \in K, d \in D \quad (9)$$

$$\sum_{k \in K} v_i^{kd} = q_i^b \quad \forall i \in N_b, k \in K, d \in D \quad (10)$$

$$v_l^{kd} = 0 \quad \forall l \in N \setminus N_b, k \in K, d \in D \quad (11)$$

$$w_l^{kd} = 0 \quad \forall l \in N \setminus N_s, k \in K, d \in D \quad (12)$$

$$\sum_{j \in N_s} r_j y_j \leq R \quad (13)$$

$$\sum_{i \in N_a} q_i^a x_{ij}^d + \sum_{k \in K} w_j^{kd} \leq C_j y_j \quad \forall j \in N_s, d \in D \quad (14)$$

$$g_{lm}^{kd} \geq 0 \quad \forall (l,m) \in A_d, k \in K, d \in D \quad (15)$$

$$v_l^{kd} \geq 0 \quad \forall l \in N, k \in K, d \in D \quad (16)$$

$$w_l^{kd} \geq 0 \quad \forall l \in N, k \in K, d \in D \quad (17)$$

$$y_j \in \{0, 1\} \quad \forall j \in N_s \quad (18)$$

$$x_{ij}^d \in \{0, 1\} \quad \forall i \in N_a, j \in N_s, d \in D \quad (19)$$

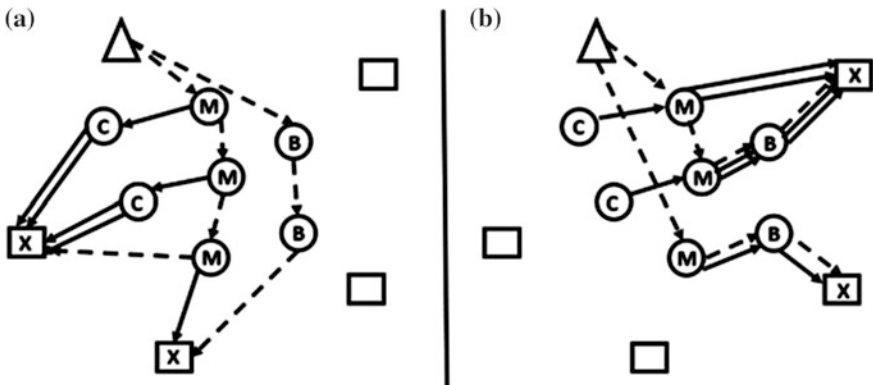
$$z_{lm}^{kd} \in \{0, 1\} \quad \forall (l,m) \in A_d, k \in K, d \in D \quad (20)$$

$$\gamma_d \geq 0 \quad \forall d \in D \quad (21)$$

The objective function (1) minimizes the expected bus-based evacuation completion time over the different network scenarios. Constraints (2) guarantee that  $\gamma_d$  is the completion time (i.e., the longest bus route) in scenario  $d$ . Constraints (3)–(5) model the car-based evacuation. For each scenario  $d$ , constraints (3) ensure that

every evacuation zone  $i \in N_a$  is assigned to exactly one shelter; constraints (4) state that evacuees can only be assigned to open shelters; constraints (5) ensure that the evacuation time for cars does not exceed a given threshold. Constraints (6)–(12) model the bus-based evacuation. For each bus  $k$  and scenario  $d$ , constraints (6) ensure that each bus departs from the depot  $o$  (if it departs); constraints (7) and (8) are flow conservation constraints; constraints (9) impose that people travel an arc only if a bus, whose capacity cannot be exceeded, serves that arc; constraints (10) guarantee that all the people of zone  $i \in N_b$  evacuate to some shelter; constraints (11) and (12) state that evacuation starts only at nodes  $i \in N_b$  and ends at shelter sites  $j \in N_s$ , respectively. Constraint (13) states that the total amount of resources available to set up shelters cannot be exceeded while constraints (14) link together the self- and supported-evacuation variables by imposing that the shelter capacity cannot be exceeded. Constraints (15)–(21) are non-negativity and binary constraints.

Note that the use of the parameter  $\alpha$  in constraints (5) allows the model to identify trade-off solutions which balance the bus-evacuation completion time objective and the self-evacuee equity requirement. An example is shown in Fig. 1: a tight threshold (Fig. 1a) favors self-evacuation by forcing the selection of shelters close to car-based or mixed evacuation zones; if the threshold is relaxed (Fig. 1b), the bus-evacuation completion time (longest bus route) improves at the expense of a longer self-evacuee travel time.



**Fig. 1** Car-based oriented solution (a) and bus-based oriented solution (b): Triangle, square and round shapes represent, respectively, the depot, candidate shelter sites and evacuation zones. Selected shelter sites are marked with a cross and different evacuation zones are identified with capital letters (i.e., C = car-based demand, B = bus-based demand, and M = mixed demand, i.e., both car- and bus-based demands). Normal and dashed arrows represent, respectively, car-evacuees assignment and bus routes

### 4 Computational Results

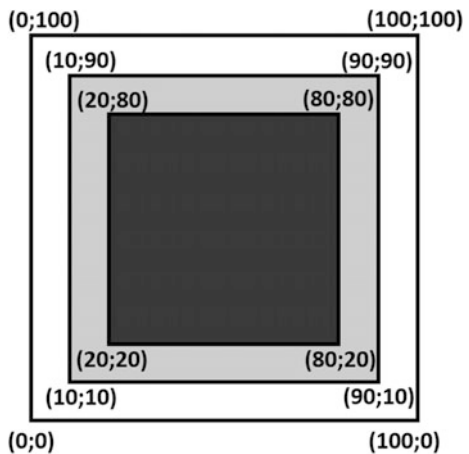
The SISLER model is a mixed-integer linear programming (MILP) model which can be solved by off-the-shelf optimization software for modest size instances. The model was implemented using IBM ILOG OPL modeling language and solved with the solver CPLEX 12.6.2 on a computer with an Intel® Core™ i5-5200U CPU @ 2.20 GHz and 8.00 GB of RAM.

SISLER was tested on small random instances generated as follows. We considered a  $100 \times 100$  square study area like the one displayed in Fig. 2, where the black area represents the central zone of the disaster and the white area the safety zone. The coordinates of evacuation zones, transshipment nodes, and shelter sites were generated at random in the black, light grey, and white areas, respectively.

Arcs were also generated at random and Euclidean distances were used as a proxy for travel times. In accordance with our contraflow model assumption, we assumed that arcs from transshipment nodes to shelter nodes can be traveled only in one direction. We considered three scenarios: (1) a base scenario, where all network arcs are available, (2) an average scenario, where some network arcs connecting evacuation nodes are not available, and (3) a worse-case scenario where some arcs between evacuation and transshipment nodes (from black to grey area) are also unavailable.

In terms of model parameters: (i) a homogeneous bus fleet was assumed and the number of buses was computed as the rounded ratio [total bus-based demand/bus capacity]; (ii) shelter capacities were computed as in [7]; (iii) for each scenario and car-based evacuation zone, the shortest travel time and the time threshold were computed in a pre-processing phase (the former through a shortest path algorithm, the latter by solving a capacitated p-center model); (iv) a decreasing probability distribution was used for the three scenarios ( $p_1 = 0.5$ ,  $p_2 = 0.3$ , and  $p_3 = 0.2$ ).

Fig. 2 Study area





**Table 1** Computational results

$\alpha$	Scenario 1			Scenario 2			Scenario 3			All scenarios		
	Bus max time	Car tot time	Shelters	Bus max time	Car tot time	Shelters	Bus max time	Car tot time	Shelters	Bus max time	Car tot time	Shelters
0	212	282	{19,20,23,24}	175	347	{19,21,23,24}	211	447	{19,20,22,24}	216	1027	{19,20,23,24}
0.1	212	282	{19,20,23,24}	175	347	{19,21,23,24}	211	447	{19,20,22,24}	216	1027	{19,20,23,24}
0.2	143	301	{19,21,23,24}	175	347	{19,21,23,24}	211	447	{19,20,22,24}	166.2	1161	{20,21,23,24}
0.3	143	301	{19,21,23,24}	175	347	{19,21,23,24}	211	447	{19,20,22,24}	166.2	1161	{20,21,23,24}
0.4	143	301	{19,21,23,24}	175	347	{19,21,23,24}	211	447	{19,20,22,24}	166.2	1161	{20,21,23,24}
0.5	138	323	{19,22,23,24}	168	427	{20,22,23,24}	211	447	{19,20,22,24}	161.6	1372	{20,22,23,24}
0.6	138	323	{19,22,23,24}	168	427	{20,22,23,24}	211	447	{19,20,22,24}	161.6	1372	{20,22,23,24}
0.7	138	323	{19,22,23,24}	168	427	{20,22,23,24}	211	447	{19,20,22,24}	161.6	1372	{20,22,23,24}
0.8	138	323	{19,22,23,24}	168	421	{19,20,22,24}	211	447	{19,20,22,24}	161.6	1367	{20,22,23,24}
0.9	138	323	{19,22,23,24}	168	421	{19,20,22,24}	211	447	{19,20,22,24}	161.6	1367	{20,22,23,24}
1	138	323	{19,22,23,24}	168	421	{19,20,22,24}	211	447	{19,20,22,24}	161.6	1367	{20,22,23,24}

The results for a network with 25 nodes and 56 arcs are displayed in Table 1. For each individual scenario and for the combined scenarios, the table reports the bus-evacuation completion time, the total self-evacuation time and the open shelters for different values of  $\alpha$  ranging from 0 to 1 (note that the potential shelter sites are nodes 19, 20, 21, 22, 23, 24).

From the analysis of the table, it is possible to infer the trade-off between the bus-based and the car-based evacuation time. For example, in the base scenario, when  $\alpha$  increases from 0.1 to 0.2, the bus-evacuation time drops by nearly 33% (from 212 to 143), while the car evacuation time increases by around 7%. Another change in both shelter locations and times can be observed for  $\alpha = 0.5$ . Note that SISLER's objective can yield multiple optimal solutions, some of which are inefficient from the car-evacuation perspective. To guarantee an efficient allocation of car-evacuees to shelters, we employed a lexicographic objective function by adding a second term (total car evacuation time) to the objective (1) (see for example [3]).

The results in the table also highlight the importance of considering multiple scenarios. The solutions found when all three scenarios are taken into account can differ quite significantly from the solutions obtained for a single scenario. For example, the optimal set of shelters in the solution obtained with  $\alpha = 0.2$  (20, 21, 23, 24) is different from the optimal set selected in each individual scenario for the same value of  $\alpha$  (and so are the bus routes and self-evacuee allocations).

## 5 Conclusions

This paper introduced a scenario-based location-allocation-routing model to optimize evacuation planning decisions. The model integrates shelter location and evacuation routing decisions, while considering both a user perspective (self-evacuation) and a system perspective (supported-evacuation). It also addresses the uncertainty of the infrastructure availability after a disaster by optimizing evacuation plans across several disruption scenarios. We demonstrated how the model can be used to identify user-system trade-off solutions on a small sample network. The example also highlights the importance of considering different disruption scenarios.

In the future, we plan to test the model on larger networks and for different probability distributions. The simple network used in this study as a proof of concept was solved by a general-purpose optimization solver in a matter of seconds. Undoubtedly, solving larger problems with many disruption scenarios will require devising ad hoc solution methods. Specifically, advanced methods for generating realistic scenarios and solving large-scale stochastic programs (e.g., Sample Average Approximation) should be developed.

The proposed model is still far from being comprehensive and could be further extended to include other complicating aspects, such as a time perspective,

congestion issues, multiple objectives, demand uncertainties and evacuee behavior. Decisions about the timing of evacuation orders and the distribution of relief supplies to shelters could also be integrated into the model.

## References

1. Altay, N., Green, W.G.: OR/MS research in disaster operations management. *Eur. J. Oper. Res.* **175**(1), 475–493 (2006)
2. Bayram, V., Tanselm, B.Ç., Yaman, H.: Compromising system and user interests in shelter location and evacuation planning. *Transport. Res. B-Meth.* **72**, 146–163 (2015)
3. Bish, D.R.: Planning for a bus-based evacuation. *OR Spectr.* **33**(3), 629–654 (2011)
4. Brachman, M., Church, R.: Planning for Disaster: A Review of the Literature With a Focus on Transportation Related Issues (First report). Geotrans Laboratory, UCSB, Santa Barbara CA (2009)
5. Chen, Z., Chen, X., Li, Q., Chen, J.: The temporal hierarchy of shelters: a hierarchical location model for earthquake-shelter planning. *Int. J. Geogr. Inf. Sci.* **27**(8), 1612–1630 (2013)
6. Cova, T.J., Johnson, J.P.: A network flow model for lane-based evacuation routing. *Transport. Res. A-Pol.* **37**(7), 579–604 (2003)
7. Gama, M., Santos, B.F., Scaparra, M.P.: A multi-period shelter location-allocation model with evacuation orders for flood disasters. *EURO J. Comput. Opt.* **4**(3–4), 299–323 (2016)
8. Goerigk, M., Grün, B., Heßler, P.: Branch and bound algorithms for the bus evacuation problem. *Comput. Oper. Res.* **40**(12), 3010–3020 (2013)
9. Goerigk, M., Deghdak, K., Heßler, P.: A comprehensive evacuation planning model and genetic solution algorithm. *Transport. Res. E-Log.* **71**, 82–97 (2014)
10. Goerigk, M., Grün, B.: A robust bus evacuation model with delayed scenario information. *OR Spectr.* **36**(4), 923–948 (2014)
11. Goerigk, M., Grün, B., Heßler, P.: Combining bus evacuation with location decisions: a branch-and-price approach. *Transport. Res. Procedia* **2**, 783–791 (2014)
12. Goerigk, M., Deghdak, K., T'Kindt, V.: A two-stage robustness approach to evacuation planning with buses. *Transport. Res. B-Meth.* **78**, 66–82 (2015)
13. Guha-Sapir, D., Below, R., Hoyois, P.: EM-DAT: The CRED/OFDA International Disaster Database—Université Catholique de Louvain—Brussels—Belgium. <http://www.emdat.be>
14. Heßler, P., Hamacher, H.W.: Sink location to find optimal shelters in evacuation planning. *EURO J. Comput. Opt.* **4**(3–4), 325–347 (2016)
15. Kongsomsaksakul, S., Yang, C., Chen, A.: Shelter location-allocation model for flood evacuation planning. *J. East. Asia Soc. Transport. Stud.* **6**, 4237–4252 (2005)
16. Li, A.C., Nozick, L., Xu, N., Davidson, R.: Shelter location and transportation planning under hurricane conditions. *Transport. Res. E-Log.* **48**(4), 715–729 (2012)
17. Murray-Tuite, P., Wolshon, B.: Evacuation transportation modeling: an overview of research, development, and practice. *Transport. Res. C-Emer.* **27**, 25–45 (2013)
18. Özdamar, L., Ertem, M.A.: Models, solutions and enabling technologies in humanitarian logistics. *Eur. J. Oper. Res.* **244**(1), 55–65 (2015)
19. Van Wassenhove, L.N.: Humanitarian aid logistics: supply chain management in high gear. *J. Oper. Res. Soc.* **57**(5), 475–489 (2006)

# The Vehicle Routing Problem with Occasional Drivers and Time Windows

Giusy Macrina, Luigi Di Puglia Pugliese, Francesca Guerriero and Demetrio Laganà

**Abstract** In this paper, we study a variant of the Vehicle Routing Problem with Time Windows in which the crowd-shipping is considered. We suppose that the transportation company can make the deliveries by using its own fleet composed of capacitated vehicles and also some occasional drivers. The latter can use their own vehicle to make either a single delivery or multiple deliveries, for a small compensation. We introduce two innovative and realistic aspects: the first one is that we consider the time windows for both the customers and the occasional drivers; the second one is the possibility for the occasional driver to make multiple deliveries. We consider two different scenarios, in particular, in the first one multiple deliveries are allowed for each occasional driver, in the second one the split delivery policy is introduced. We propose and validate two different mathematical models to describe this interesting new setting, by considering several realistic scenarios. The results show that the transportation company can achieve important advantages by employing the occasional drivers, which become more significant if the multiple delivery and the split delivery policy are both considered.

**Keywords** Vehicle routing problem · Crowd-shipping · Occasional drivers

---

G. Macrina (✉) · L. Di Puglia Pugliese · F. Guerriero · D. Laganà  
Department of Mechanical, Energy and Management Engineering,  
University of Calabria, 87036 Rende (CS), Italy  
e-mail: giusy.macrina@unical.it

L. Di Puglia Pugliese  
e-mail: luigi.dipugliapugliese@unical.it

F. Guerriero  
e-mail: francesca.guerriero@unical.it

D. Laganà  
e-mail: demetrio.lagana@unical.it

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_58

## 1 Introduction

In the last years the growing importance of shorter delivery lead times has led the companies to create innovative solutions to organize the last-mile and same-day delivery. In this context, the “sharing economy” has attracted a great deal of interest. Sharing assets and capacities can enhance the use of resource and become a new opportunity to pursue the efficiency in transportation issue. One of the innovative solution is the crowd-shipping, i.e. ordinary people bring items for other people en-route to their destination. The rapid growth in on-line retailing has encouraged the retailers to develop innovative solutions of last-mile delivery. Walmart, DHL and Amazon are among those big retailers who started to use the crowd-shipping and its potential. Walmart, in 2013, announced it was working on a plan to outsource some of its deliveries to its on-line customers.

“MayWays” is the pilot last-mile crowd-shipping service of DHL in Stockholm. Thus, people in Stockholm, mostly students, can use a smartphone app in order to see the uploaded requests. Whereupon, they can decide if they want to pick up the package at a DHL facility and deliver it to the final destination. In 2015, Amazon launched its new service of crowd-shipping, called Amazon Flex, and nowadays it is already used in more than 30 cities in the world. People use the Amazon Flex app to become a delivery partner and set their own schedule.

In this context, crowd-shipping seems to be a new transport solution that offers several potential benefits. The crowd-shipping strategy exploits the personal vehicles capacity that usually travel on roads. This can be useful to reduce the need for separate freight deliveries and it allows to exploit the whole capacity of a vehicle, that is not fully used. There is also a “green” aspect that can be taken into account, in particular, the sharing of the vehicles can lead to the reduction of the pollutant emissions, the energy consumption, the noise and the traffic. In Arslan et al. [3], the authors analyze the potential benefits of crowd-sourced delivery. They present a complete literature review of the recent contributions dealing with crowd-sourcing. They consider a peer-to-peer platform, taking into account the possibility to use both traditional vehicles and ad-hoc vehicles. They also present a rolling horizon framework and an exact solution approach to solve the routing planning problem. In this work, we propose a variant of the Vehicle Routing Problem with Time Windows (VRPTW), starting from the work presented in Archetti et al. [1], in which the crowd-shipping is considered. In Archetti et al. [1], the authors propose a new problem called VRPOD (Vehicle Routing Problem with Occasional Drivers). In the VRPOD, the transportation company can make the deliveries not only by using its own fleet composed by capacitated vehicles, but also by making use of the services of some occasional drivers (ODs). The latter can use their own vehicle to make a single delivery, for a small compensation calculated by evaluating the deviation from their predefined route. They propose an integer programming formulation for the VRPOD and develop a multi-start heuristic, which combines variable neighbourhood search and tabu search. We introduce two innovative aspects to the problem proposed in Archetti et al. [1]. The first one is that we consider time windows constraints for

both the customers and the occasional drivers (VRPODTW); indeed, it is merely improbable that an occasional driver is all the time available to make the delivery to the customers. The second one is the possibility for the ODs to make not only a single delivery. As a matter of fact, if on the occasional driver's way there is more than one customer to be served, and the constraints related to time and load are satisfied, the multiple delivery is allowed. The proposed mathematical models are described in Sect. 2. The computational experiments are presented in Sect. 3. Finally, Sect. 3.3 summarizes the conclusions.

## 2 The Vehicle Routing Problem with Occasional Drivers

We model the problem on a complete directed graph  $G = (N, A)$ , with node set  $N = C \cup \{s, t\} \cup V$ , where  $C$  is the set of customers while  $s$  is the origin node and  $t$  is destination node for the classic vehicles. Let  $A$  be the set of arcs.  $K$  is the set of available ODs while  $V$  is the set of  $v_k$  destinations associated with the ODs. Each arc  $(i, j) \in A$  has a cost  $c_{ij}$  and a time  $t_{ij}$  associated with it. Note that both  $c_{ij}$  and  $t_{ij}$  satisfy the triangle inequality. Each node  $i \in C \cup V$  has a time windows defined as  $[e_i, l_i]$ . Each customer  $i \in C$  has a demand  $d_i$ .  $Q$  is the capacity of the classic vehicles,  $P$  is the number of available classic vehicles, while  $Q_k$  is the capacity of OD  $k \in K$ . Let  $x_{ij}$  be a binary variable that is equal to 1 if a classical vehicle traverses the arc  $(i, j)$ , and 0 otherwise. For each node  $i \in N$  let  $y_i$  be the available capacity of the vehicle after visiting customer  $i$ , while  $s_i$  is the arrival time of the vehicle to the customer  $i$ . Moreover  $r_{ij}^k$  is a binary variable that is equal to 1 if the OD  $k$  traverses the arc  $(i, j)$ , 0 otherwise. Let  $f_i^k$  indicate the arrival time of OD  $k$  to the customer  $i$  and let  $w_i^k$  be the available capacity of OD  $k$  after visiting customer  $i$ . At first we consider the scenario in which multiple deliveries for the OD are allowed and we called this version: *VRPODTWmd*. The *VRPODTWmd* can be formulated as follows:

$$\text{Min } \sum_{i \in C \cup \{s\}} \sum_{j \in C \cup \{t\}} c_{ij} x_{ij} + \sum_{k \in K} \sum_{i \in C \cup \{s\}} \sum_{j \in C} \rho c_{ij} r_{ij}^k - \sum_{k \in K} \sum_{j \in C} c_{sv_k} r_{sj}^k \tag{1}$$

$$s.t. \sum_{j \in C \cup \{t\}} x_{ij} - \sum_{j \in C \cup \{s\}} x_{ji} = 0, \quad \forall i \in C \tag{2}$$

$$\sum_{j \in C} x_{sj} - \sum_{j \in C} x_{jt} = 0 \tag{3}$$

$$y_j \geq y_i + d_j x_{ij} - Q(1 - x_{ij}), \quad \forall j \in C \cup \{t\}, i \in C \cup \{s\} \tag{4}$$

$$y_s \leq Q \tag{5}$$

$$s_j \geq s_i + t_{ij} x_{ij} - \alpha(1 - x_{ij}), \quad \forall i \in C, j \in C \tag{6}$$

$$e_i \leq s_i \leq l_i, \quad \forall i \in C \tag{7}$$

$$\sum_{j \in C} x_{sj} \leq P \tag{8}$$

$$\sum_{j \in CU\{v_k\}} r_{ij}^k - \sum_{h \in CU\{s\}} r_{hi}^k = 0, \quad \forall i \in C, k \in K \quad (9)$$

$$\sum_{j \in CU\{v_k\}} r_{sj}^k - \sum_{j \in CU\{s\}} r_{jv_k}^k = 0, \quad \forall k \in K \quad (10)$$

$$\sum_{k \in K} \sum_{j \in CU\{v_k\}} r_{sj}^k \leq |K| \quad (11)$$

$$\sum_{j \in C} r_{sj}^k \leq 1, \quad \forall k \in K \quad (12)$$

$$w_j^k \geq w_i^k + d_i r_{ij}^k - Q_k(1 - r_{ij}^k), \quad \forall j \in CU\{v_k\}, i \in CU\{s\}, k \in K \quad (13)$$

$$w_s^k \leq Q_k, \quad \forall k \in K \quad (14)$$

$$f_i^k + t_{ij} r_{ij}^k - \alpha(1 - r_{ij}^k) \leq f_j^k, \quad \forall i \in C, j \in C, k \in K \quad (15)$$

$$f_i^k \geq e_{v_k} + t_{si}, \quad \forall i \in C, k \in K \quad (16)$$

$$f_{v_k}^k \leq l_{v_k}, \quad \forall k \in K \quad (17)$$

$$f_i^k + t_{iv_k} r_{iv_k}^k - \alpha(1 - r_{iv_k}^k) \leq f_{v_k}^k, \quad \forall i \in C, k \in K \quad (18)$$

$$e_i \leq f_i^k \leq l_i, \quad \forall i \in C \quad (19)$$

$$\sum_{j \in CU\{t\}} x_{ij} + \sum_{h \in CU\{v_k\}} \sum_{k \in K} r_{ih}^k = 1, \quad \forall i \in C \quad (20)$$

$$x_{ij} \in \{0, 1\}, \quad \forall (i, j) \in A \quad (21)$$

$$r_{ij}^k \in \{0, 1\}, \quad \forall (i, j) \in A, k \in K \quad (22)$$

$$0 \leq y_i \leq Q, \quad \forall i \in CU\{s, t\} \quad (23)$$

$$0 \leq w_i^k \leq Q_k, \quad \forall i \in CU\{s, v_k\}, k \in K \quad (24)$$

$$f_i^k \geq 0, \quad \forall i \in CU\{s, v_k\}, k \in K \quad (25)$$

The objective function 1 aims to minimize the total costs. The first term is the transportation cost associated with the vehicles. The second term is the cost of compensation of the OD  $k$  for the delivery service with  $\rho \geq 0$ , the third one is the cost of the OD  $k$  when it does not perform the delivery service. Constraints 2–8 are linked to the classical vehicles. Constraint 2–3 are the flow constraints. Constraints 4 guarantee the fulfilment of demand at customer vertices. Constraints 5 restrict the initial cargo load level to the maximum capacity of a vehicle. Constraints 6 allow to determine the arrival time at node  $j$ , while constraints 7 guarantee arrival within the time window at each node. Constraints 8 impose a maximum number of available vehicles. Constraints 9–19 are linked to the ODs. Constraint 9–10 are the flow constraints. Constraints 11–12 guarantee a limit on the number of available ODs and the number of departs from the depot. Constraints 13–14 are the capacity constraints. Constraints 15 allow to determine the arrival time at node  $j$ . Constraints 16–17 are the time windows constraints and they also define the time in which the ODs are available to make the deliveries, while constraints 18 allow to determine the arrival

time at the destination node  $v_k$ . Constraints 19 assure that each customer is served within its time windows. Constraint 20 guarantees that each customer is visited at most once, by either a classic vehicle or an OD. We also formulate a second *VRPOD* variant, called *VRPODTWsd* in which we consider a split delivery policy for the ODs. Thus, the assumption that each customer is visited only once by the ODs is relaxed (constraint 20). We introduce a new variable  $o_i^k$  that indicates the quantity of demand  $d_i$  delivered by the OD  $k \in K$  to the customer  $i \in C$ . In order to define the *VRPODTWsd*, starting from *VRPODTWmd*, constraints 13 and 20 are modified as follows:

$$w_j^k \geq w_i^k + o_i^k - Q_k(1 - r_{ij}^k), \quad \forall j \in C \cup \{v_k\}, i \in C \cup \{s\}, k \in K \quad (26)$$

$$\sum_{j \in C \cup \{t\}} x_{ij} + \sum_{h \in C \setminus \{v_k\}} \sum_{k \in K} r_{ih}^k \geq 1, \quad \forall i \in C \quad (27)$$

It is also necessary to introduce new constraints for modelling the split delivery policy for the ODs:

$$\sum_{k \in K} o_i^k + d_i \sum_{j \in C \cup \{s\}} x_{ji} = d_i, \quad \forall i \in C \quad (28)$$

$$\sum_{i \in C} o_i^k \leq Q_k, \quad \forall k \in K \quad (29)$$

$$o_i^k \geq 0, \quad \forall i \in C, k \in K \quad (30)$$

### 3 Computational Experiments

This section presents the results of computational experiments performed in order to validate the proposed models. The main goal is to demonstrate the potential benefits that can be obtained by using the crowd-shipping in a realistic scenario. With this purpose we take into account the state-of-art mathematical model proposed in Archetti et al. [1], we add to this problem the time windows (*VRPODTW*) and we find the optimal solution by solving it with a commercial solver. Whereupon, we solve our proposed models and compare the obtained results. We divided the comparative analysis into two phases, in the first one the results obtained solving the *VRPODTW* is compared to those obtained by solving the *VRPODTWmd*. In the second phase, we present a comparative analysis of the results obtained by solving the *VRPODTWmd* and those obtained with the split delivery policy, the *VRPODTWsd*. The models were implemented and solved with the commercial solver CPLEX 12.5 and run on a computer with an Intel Core i5 processor at 2.70 GHz and 4 GB of RAM. We first describe the generated instances and after the results.



### 3.1 Generation of VRPOD Instances

The instances, used to assess the behaviour of proposed models in terms of solutions quality, are based on the classical Solomon VRPTW instances (see Solomon [2]). As well known, these instances are divided into 3 classes C, R and RC that differ for the geographical distribution of the customer locations: a clustered distribution (C), random distribution (R) and a mix of both (RC). Each class is divided into two subclasses, the first one (C1, R1, RC1) has a short scheduling horizon, while the second one (C2, R2, RC2) has a long scheduling horizon. We create a set of 36 small instances randomly choosing 5, 10 and 15 customers and 3 OD destinations. To obtain the problem tests for the VRPOD, given a VRPTW instance with the customers locations identified by the coordinates  $(x_i, y_i)$ , we randomly generate 3 destinations for the ODs, in the square with lower left hand corner  $(min_i x_i, min_i y_i)$  and upper right hand corner  $(max_i x_i, max_i y_i)$ , (see Archetti et al. [1]). After we randomly generate a reasonably time window.

### 3.2 Comparative Analysis

We now present a comparative analysis of the results, divided into two phases. At first we take into account the results of the VRPODTW and those obtained by solving the VRPODTWmd, whereupon, we introduce the results obtained solving the VRPODTWsd. We use the settings reported in the Tables 1 and 3 for the first part of experiments and for the second one, respectively. Table 2 presents the comparison results for each VRPODTW instance against VRPODTWmd. Each table has 4 columns, for each version of the model. The first one shows the name of the instance, the second one the cost of the solution, in the third one #CD is the number of the classical drivers, while #OD, in the fourth one, the number of the ODs. In the last column the “GAP” on the cost is calculated as follows:  $Gap = \frac{ObjVRPOD - ObjVRPODTWmd}{ObjVRPOD}$ .

The computational results show the VRPODTWmd model outperforms VRPODTW in terms of solution quality. The “Gap” is equal to 13% for the instances with 5 and 15 node and 10% for the instances with 10 customers. The reduction of the cost is due to the use of the ODs, that allowed to make multiple deliveries. The

**Table 1** Parameters setting

# Customers	Parameters							
	C	P	Q	K	Q <sub>1</sub>	Q <sub>2</sub>	Q <sub>3</sub>	ρ
5	5	1	100.0	3	30.0	30.0	40.0	1.1
10	10	2	100.0	3	30.0	30.0	40.0	1.1
15	15	3	100.0	3	40.0	40.0	60.0	1.1

**Table 2** Results for the VRPODTW and VRPODTWmd

(a) Results for instances with 5 customers

Test	VRPODTW			VRPODTWmd			
	Cost	# CD	# OD	Cost	# CD	# OD	Gap%
C101C5	139.7	1	2	124.9	1	2	11
C103C5	110.3	1	3	106.6	1	2	3
C206C5	159.6	1	1	138.4	1	2	13
C208C5	113.3	1	2	91.1	0	3	20
R104C5	83	1	2	83	1	2	0
R105C5	91.5	1	3	77.5	1	3	15
R202C5	125.8	1	2	125.8	1	2	0
R203C5	125.3	1	3	93.4	1	3	25
RC105C5	134	1	2	126.6	1	2	6
RC108C5	210.5	1	2	164	1	2	22
RC204C5	107	1	2	107	1	2	0
RC208C5	122.6	1	3	76	1	3	38
<b>Avg</b>	126.8833			109.525			<b>13</b>

(b) Results for instances with 10 customers

Test	VRPODTW			VRPODTWmd			
	Cost	# CD	# OD	Cost	# CD	# OD	Gap%
C101C10	261.7	2	3	250.6	2	3	4
C104C10	223.7	2	3	196.9	1	3	12
C202C10	181.4	2	3	172.3	1	3	5
C205C10	184.7	2	2	172.9	1	3	6
R102C10	169.7	2	3	134.0	1	3	21
R103C10	150.0	2	0	122.7	1	2	18
R201C10	154.6	2	3	146.3	2	3	5
R203C10	149.7	1	3	132.9	1	3	11
RC102C10	294.2	2	3	276.0	1	3	6
RC108C10	276.2	2	3	235.2	2	2	15
RC201C10	242.8	2	2	226.3	1	2	7
RC205C10	270.1	2	2	260.3	2	2	4
<b>Avg</b>	213.2333			193.8667			<b>10</b>

(continued)

**Table 2** (continued)

(c) Results for instances with 15 customers

Test	VRPODTW			VRPODTWmd			
	Cost	# CD	# OD	Cost	# CD	# OD	Gap%
C103C15	296.6	2	2	264.8	2	1	11
C106C15	222.4	2	2	173.8	1	3	22
C202C15	322.0	3	2	285.2	2	2	11
C208C15	270.9	3	1	255.9	2	2	6
R102C15	305.4	3	2	279.9	2	2	8
R105C15	279.6	3	2	244.0	2	2	13
R202C15	345.4	3	1	320.9	2	2	7
R209C15	276.4	3	1	240.1	2	2	13
RC103C15	336.3	3	2	248.4	2	2	26
RC108C15	377.0	3	1	334.8	2	2	11
RC202C15	362.5	3	1	281.5	2	2	22
RC204C15	357.4	3	2	326.6	2	2	9
<b>Avg</b>	312.7			271.3			<b>13</b>

**Table 3** Parameters setting

# cus-tomers	Parameters							
	$ C $	$P$	$Q$	$ K $	$Q_1$	$Q_2$	$Q_3$	$\rho$
5	5	2	60.0	3	5.0	10.0	15.0	1.1
10	10	2	75.0	3	10.0	10.0	15.0	1.1
15	15	3	75.0	3	10.0	15.0	15.0	1.1

solutions of the VRPODTW model use a number of classical drivers greater than the one used in the VRPODTWmd’s solutions (35% more) and the cost of the solution is higher. While, solving VRPODTWmd allows to involve the 9.52% of ODs more than VRPODTW. However, it is possible to highlight that, even if the same configuration of vehicles is obtained, the solutions obtained with VRPODTWms are more competitive than those obtained with VRPODTW. E.g. in the solutions of the instance “RC208C5” both the models consider one classical driver and two occasional drivers, however, the cost for the VRPODTW solution is about the 60% higher. Overall, the use of ODs allowed to make multiple deliveries optimizes the total costs and reduces the use of classical vehicles.

The Table 4 present the comparison results for each VRPODTWmd instance against VRPODTWsd. The results of Table 4 clearly underline that the use of the split delivery strategy results competitive in terms of effectiveness. On average, a cost reduction of about 10% is observed. The possibility to split the deliveries increases the number of ODs used by the VRPODTWsd, with a consequent cost saving, i.e.

**Table 4** Results for the VRPODTW<sub>md</sub> and VRPODTW<sub>sd</sub>

**(a)** Results for instances with 5 customers

Test	VRPODTW <sub>md</sub>			VRPODTW <sub>sd</sub>			
	Cost	# CD	# OD	Cost	# CD	# OD	Gap%
C101C5	213.1	2	1	195.3	2	2	8
C103C5	159.0	2	0	153	2	2	4
C206C5	202.3	1	1	175.4	1	2	13
C208C5	154.2	1	2	141.4	1	2	8
R104C5	166.0	2	0	162.3	2	2	2
R105C5	155.1	2	1	149.3	2	3	4
R202C5	170.0	2	0	156.8	2	2	8
R203C5	179.0	2	1	174.2	1	3	3
RC105C5	228.0	2	0	177.1	2	2	22
RC108C5	203.5	1	1	202.5	1	2	0
RC204C5	173	2	1	118.3	1	3	32
RC208C5	189.8	2	1	174.2	2	3	8
<b>Avg</b>	182.8			164.9			<b>9</b>

**(b)** Results for instances with 10 customers

Test	VRPODTW <sub>md</sub>			VRPODTW <sub>sd</sub>			
	Cost	# CD	# OD	Cost	# CD	# OD	Gap%
C101C10	353.3	2	3	324.7	2	3	8
C104C10	315.8	2	3	276	2	3	13
C202C10	295.9	2	3	227.8	2	3	23
C205C10	206.2	2	1	188	2	2	9
R102C10	208.9	2	1	198.6	2	3	5
R103C10	185.7	2	2	167	2	3	10
R201C10	263.8	2	2	209.6	2	3	21
R203C10	204.7	2	1	172.3	2	3	16
RC108C10	434.6	2	3	393.9	2	3	9
RC102C10	396.6	2	2	391	2	3	1
RC201C10	333.4	2	2	327.2	2	3	2
RC205C10	340.8	2	2	340.8	2	2	0
<b>Avg</b>	294.9			268.1			<b>10</b>

(continued)

**Table 4** (continued)

(c) Results for instances with 5 customers

Test	VRPODTWmd			VRPODTWsd			
	Cost	# CD	# OD	Cost	# CD	# OD	Gap%
C103C15	318.4	3	1	318.4	3	1	0
C106C15	253.3	3	2	247.8	3	2	2
C202C15	413.6	3	3	404.8	3	2	2
C208C15	154.2	1	2	141.4	1	2	8
R102C15	360.8	3	2	315.6	3	3	13
R105C15	309.6	3	2	299.6	3	3	3
R202C15	392.4	3	2	379.2	3	2	3
R209C15	345.5	3	2	328.6	3	2	5
RC103C15	360.9	3	2	356.9	3	2	1
RC108C15	488.2	3	3	465.5	3	3	5
RC202C15	479.9	3	3	444.5	3	3	7
RC204C15	346.0	3	0	345.7	3	2	0
<b>Avg</b>	351.9			337.3			<b>4</b>

the 62.50% more than those used by the VRPODTWmd. Also for these experiments, when the same configuration of vehicles is used in the solutions, often VRPODTWsd outperforms VRPODTWmd. E.g. for the instances “C208C5”, “C101C10” and “C208C15” in which the same number of classical and occasional drivers are used in the solutions, the “Gap” is equal to 8%. In summary, the presented models outperform the literature model in terms of effectiveness. The computational experiments highlight the benefits reached when the ODs are used to make deliveries, which become more interesting when the split delivery policy is considered.

### 3.3 Conclusions

We have proposed two innovative variants for the VRP. We take into account the possibility that a company may use the service provided by some ODs. The ODs are available to make some deliveries for a small compensation. The main goal has been to investigate the achievable potential benefits by introducing the crowd-sourcing in the VRP. The results of our computational experiments are very encouraging. We demonstrated that the use of the ODs may improve the routing plan, generating an interesting cost saving. The possibility to make multiple deliveries and the split delivery policy allows to exploit the whole capacity of the ODs. This work can be viewed as a base for several future works. There are more aspects that can be taken into account. For example the “green” aspect of this strategy. In fact, the use of the ODs reduces the pollutant emissions and the traffic congestion. The ODs perform

travels that ordinarily already take place, thus, there is a reduction of routed vehicles and distance travelled. There is also the possibility to deliver the goods by bicycles or public transport. In conclusion, crowd-shipping allows the company to outsource the “last mile” deliveries to ordinary citizens and this may be an opportunity but also a risk. The company may provide a convenient and efficient delivery service. However, misuse the crowd-shipping implies giving more responsibility to the ODs, and it is intrinsically a risk.

## References

1. Archetti, C., Savelsbergh, M., Speranza, M.G.: The vehicle routing problem with occasional drivers. *Eur. J. Oper. Res.* **254**, 472–480 (2016)
2. Solomon, M.M.: Algorithms for the vehicle routing and scheduling problems with time window constraints. *Oper. Res.* **35**(2), 254–265 (1987)
3. Arslan, A.M., Agatz, N., Kroon, L., Zuidwijk, R.: Crowdsourced delivery: a dynamic pickup and delivery problem with ad-hoc drivers. Erasmus Research Institute of Management ERIM paper, Social Science Research Network. [http://papers.ssrn.com/sol3/papers.cfm?abstracts\\_id=2726731](http://papers.ssrn.com/sol3/papers.cfm?abstracts_id=2726731) (2016)

# **Part XII**

## **Scheduling**

# Preemptive Scheduling of a Single Machine with Finite States to Minimize Energy Costs

Mohammad Mohsen Aghelinejad, Yassine Ouazene and Alice Yalaoui

**Abstract** This paper addresses a single machine scheduling problem in which the system may switch among three different states, namely ON (needed for processing the jobs), OFF or Idle. Each state, as well as switching among the different states, consume energy. The objective is schedule  $n$  preemptive jobs to minimize the total energy costs. Time varied electricity price are considered. The complexity of this problem is investigated using a new dynamic programming approach. In this approach, a finite graph is used to model the proposed problem. The dimension (number of vertices and edges) of this graph is dependent on the total processing times and the total number of periods. Then, the optimal solution of the problem is provided by calculating the shortest path between the first node and last node representing respectively the first and the last periods. Based on the Dijkstra's algorithm complexity, we prove that the complexity of this problem, is polynomial of degree 3.

**Keywords** Preemption scheduling problem · Time-dependent energy costs  
Dynamic programming · Dijkstra's algorithm

## 1 Introduction

Nowadays, the increase of the electricity prices in the most industrial countries attracted the attention of many researchers all around the world. A comprehensive

---

M.M. Aghelinejad (✉) · Y. Ouazene (✉) · A. Yalaoui (✉)  
Industrial Systems Optimization Laboratory (ICD, UMR 6281, CNRS),  
Université de Technologie de Troyes, 12 rue Marie Curie,  
CS 42060-10004 Troyes cedex, France  
e-mail: mohsen.aghelinejad@utt.fr

Y. Ouazene  
e-mail: yassine.ouazene@utt.fr

A. Yalaoui  
e-mail: alice.yalaoui@utt.fr



review of previous studies demonstrates that minimization of energy consumption in a manufacturing system can be applied in three different sectors: machine-level, product-level, and system-level. The study of machine-level or product-level needs enormous financial investments to study the machine(s) or product(s) redesign procedures. But, in the system-level's study, manufacturers may reduce the system's energy consumption by using the existing decision models and optimization techniques in production planning and scheduling. So, in this paper, the system-level's study is considered with some decreasing energy consumption's methods for a manufacturing system with single machine.

The total energy consumptions of a production system can be divided into the non-processing states (NPE) energy consumptions (the start-up, the transition between different states, shut down and idle states), and the processing state (PE) energy consumptions. Also, the amount of machine's energy consumption depends on the type of machine or jobs, the state of the machine, and processing speed of the machine during each state. One of the easiest and most popular ways for total energy consumption minimization is investigating the NPE consumption states and using a scheduling method to change the processing job's order and machine's state during a production shift.

In this section, we give few examples of the studies that investigated these types of problems. The complexity of a classical deadline-based scheduling problem for the non-preemptive and preemptive cases with variable speed processing are studied in [2, 4]. The multi-objective models that minimizes the energy consumption and a traditional scheduling performance measure like total completion time and total tardiness are presented in [12, 13, 15]. Different energy charging policies like time-of-use (TOU) pricing, real-time pricing, and critical peak pricing, can be considered to investigate the total energy cost of a system. An energy-conscious single machine scheduling problem, when each processing job has its power consumption, and electricity prices may vary from hour to hour throughout a day, are assumed in [3]. Energy constraint and different energy cost during the planning horizon of a flow-shop system are considered in [9, 10]. The generic mixed-integer programming models for a single machine scheduling that minimize total energy cost at volatile energy prices are presented in [6, 7]. The minimization of total electricity consumption costs and operations postponement penalty costs for a preemptive scheduling problem with energy constraint, different power demand for each job, and the electricity time-varying prices is investigated in [11]. In [14], a mathematical model is proposed to minimize total energy consumption costs for a single machine with different possible states and energy consumptions. The problem with the same assumptions is considered in [1] to improve the previous mathematical model. They also presented a new mathematical model to obtain the optimal schedule for the machine state and job's sequence simultaneously. A novel production scheduling method for minimizing the energy cost of a system with finite states machine, multiple process idle modes and time varied electricity price are addressed in [8].

The rest of this paper is organized as follows. Section 2 introduces the problem statement. Also, the different notations and assumptions are described. Section 3 proposes a dynamic programming method to define the problem. Section 4 deals with

the complexity analysis of the proposed scheduling problem. The numerical experiments results are reported in Sect. 5. Finally, Sect. 6 summarizes the contributions of this paper and some perspectives.

## 2 Problem Statement

This paper addresses the problem of scheduling  $n$  jobs on a single machine, which may switch among three different states, namely ON (needed for processing the jobs), OFF or Idle, within a given planning horizon  $T$ . When the machine is in state OFF, a fixed number of periods ( $\beta_1$ ) will elapse until the machine is ready to process a job. Likewise, when the machine is in state ON, a different fixed number of periods ( $\beta_2$ ) will elapse until it arrives in OFF state.

Each state, as well as switching among the states, entails a certain energy consumption. Energy consumption of each state is denoted by  $e_s; \forall s \in \{1, \dots, 5\}$ . These numbers are related respectively to state OFF, Turn on, ON, Idle, and Turn off, and their values are the problem's input (Fig. 1). The machine processes the jobs in state ON, and  $e_1$  (the energy consumption of machine in state OFF), is assumed to be equal to zero in this study.

Each job  $j \in \{1, \dots, n\}$  has a required process time  $p_j$ . All the jobs are available from the initial period and they can be processed preemptively during the state ON of the machine. The machine must be in state OFF for initial and final periods.

The objective of this study is to find an optimal schedule for the machine states and jobs' sequence to minimize the total energy costs of the system, when time varied electricity prices are considered during the horizon time ( $c_t$ ).

The non-preemptive version of this problem with fixed sequence is presented in [1]. So, the mathematical model of this study can be obtained by relaxing the related constraints for non-preemption and fixed sequence of the model detailed in [1].

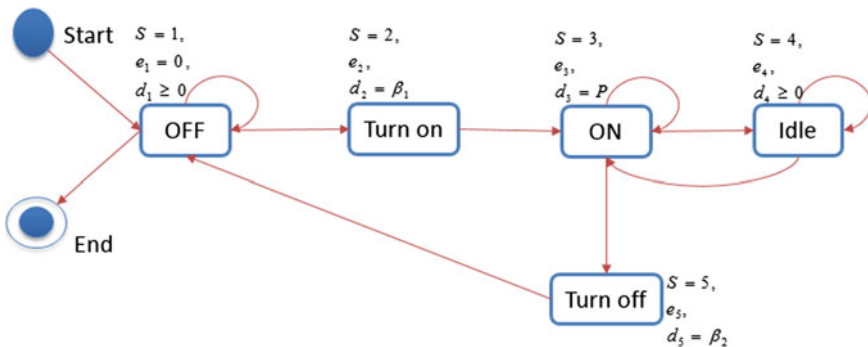


Fig. 1 Machine states and possible transitions

For the studied problem, a number of jobs must be schedule during a fixed horizon time. The decision makers are facing two categories of periods: the required periods and the surplus periods which is denoted by  $x$ . The surplus periods must be allocated to non-processing states. In our study, the non-processing states for the system are considered as the OFF state(s) at the beginning or end of the horizon, Idle state(s) and/or OFF state(s) in the middle of the horizon between the processing states. The minimum number of required periods for processing all the jobs completely, is equal to sum of the total process times ( $P = \sum_{j=1}^n p_j$ ), plus the require periods for initial turning on ( $\beta_1$ ), final turning off ( $\beta_2$ ) and 1 for the final period that machine must be in state OFF. Therefore, the extra periods within the horizon time in which the machine should be in non-processing states are equal to  $x = T - [P + (\beta_1 + \beta_2 + 1)]$ .

As a first contribution of this paper, a new dynamic programming method is proposed as will be described in the next section.

### 3 Dynamic Programming Modelling Approach

In this section, a new dynamic programming approach is developed to model all the possible solutions of this problem on a graph. This graph is composed of  $T + 1$  decision levels that indicate each period of the horizon time. Due to the extra periods at each decision level ( $t$ ), there are a set of states (nodes) ( $N_t$ ) that correspond to cumulative possible number of production units ( $k$ ) from period 0 to  $t$ . Therefore, the possible values for the states are  $\{0, \dots, P + 1\}$ , where 0 indicates the initial shut down state when any job is not processed yet, and  $P + 1$  indicates the final shut down state when all the jobs ( $P$ ) are processed. Each node may be identified by a  $(k, t)$  notation, where,  $0 \leq k \leq P + 1$  and  $0 \leq t \leq T$ . As an example, for the problem illustrated at Fig. 2,  $N_4 = \{0, 1, 2\}$ ;  $N_{10} = \{2, 3, 4, 5, 6\}$ .

As decision makers are facing  $x$  extra periods, or in the other words,  $x$  non-processing states that should be located within all the time horizon, each machine's state  $k$  has a possible set  $\{t_{min(k)}, \dots, t_{max(k)}\}$  in the time horizon.

For example, the latest period that machine can be in the initial shutdown state  $t_{max(0)}$  is when enough periods remain for performing the necessary setups and processing all the jobs. So,  $t_{max(0)} = T - \beta_1 - P - \beta_2 - 1 = x$ . The related set for node  $k$  is:

$$\{t_{min(k-1)} + \beta_{k-1,k} + 1, \dots, t_{max(k-1)} + \beta_{k-1,k} + 1\} \tag{1}$$

where,  $\beta_{k-1,k}$  is the required number of periods for transition from  $k - 1$  to  $k$ . In this study,  $\beta_{k-1,k} \forall k \in \{2, \dots, P\}$  is considered as 0. Therefore, the interval of possible positions for node  $k$  can be simplified as:

$$t_0 \in \{0, \dots, x\}; t_{p+1} \in \{T - x, \dots, T\}; t_k \in \{\beta_1 + k, \dots, x + \beta_1 + k\}; \forall k \in \{1, \dots, P\} \tag{2}$$

The edges of the graph that connect two vertices, can be divided in three main sets, and they are valued ( $V_{(k,t)-(k',t')}$ ) by the total energy cost for related transition (positive

value). The first set ( $E_1$ ), is related to the connection between two nodes with the same production level  $k$  in decision level  $t$  and  $t + 1$ . These edges can indicate the initial and final shutdown with the edge value of 0, and the idle state with the edge value of:

$$V_{(k,t)-(k,t+1)} = c_{t+1} \times e_4 \quad ; \forall k \in \{1, 2, \dots, P\} \tag{3}$$

where,  $c_t$  is the cost of energy per unit in period  $t$ , and  $e_4$  is the machine's energy consumption in idle state. The cardinal of  $E_1$  is  $|E_1| = (P + 2) \times x$ .

The second set ( $E_2$ ), is related to the connection between a node of production level  $k$  at period  $t$  with a node of production level  $k + 1$  at period  $t' > t$ . This set of edges illustrates three transitions cases:

- initial turning on with the edge value of:

$$V_{(0,t)-(1,t')} = \sum_{i=t+1}^{t'-1} (c_i \times e_2) + c_{t'} \times e_3 \quad ; \forall t' = t + \beta_1 + 1 \tag{4}$$

- processing the next job with the edge value of:

$$V_{(k,t)-(k+1,t')} = c_{t'} \times e_3 \quad ; \forall t' = t + 1 \tag{5}$$

- final turning off with the edge value of:

$$V_{(P,t)-(P+1,t')} = \sum_{i=t+1}^{t'-1} (c_i \times e_5) + c_{t'} \times e_2 \quad ; \forall t' = t + \beta_2 + 1 \tag{6}$$

The cardinal of this set of edges is equal to  $|E_2| = (P + 1) \times (x + 1)$ .

The last set of the edges ( $E_3$ ), is related to middle shutdowns between two processing parts, that connects nodes  $k$  in level  $t'$ , and node  $k + 1$  in level  $t$ . Where,  $t' \in \{t_{min(k)}, \dots, x + k - \beta_2 - 1\}$  with the edge value of:

$$V_{(k,t')-(k+1,t)} = \sum_{i=t'+1}^{t'+\beta_2} (c_i \times e_5) + \sum_{i=t-\beta_1}^{t-1} (c_i \times e_2) + c_t \times e_3 \quad ; \forall k \in \{1, 2, \dots, P - 1\} \tag{7}$$

The total number of the third set of edges is equal to:

$$|E_3| = \sum_{i=1}^{x-(\beta_1+\beta_2)} i \times (P - 1) = \frac{(x - (\beta_1 + \beta_2)) \times (x - (\beta_1 + \beta_1) + 1)}{2} \times (P - 1) \tag{8}$$

Therefore, the total number of vertices and edges for the presented graph are:

$$|V| = (P + 2) \times (x + 1) \cong TP \quad ; |E| = |E_1| + |E_2| + |E_3| \cong T^2P \tag{9}$$

To illustrate this graph construction method, we consider an example with  $P = 5, T = 15, \beta_1 = 2, \beta_2 = 1, x = 6$ , and different energy prices in horizon. The corresponding graph consists of 49 vertices and 108 edges (see Fig. 2), and the value of each edge is presented on it.

### 4 Problem Complexity Analysis

Based on this graph modelling approach, each path from the initial level at period 0 to level P+1 at period T represents a feasible solution. As the objective of this problem is the total energy costs minimization, the shortest path between these two nodes, is the optimal solution of the problem. To illustrate that this may be achieved in polynomial time, we consider Dijkstra’s algorithm. We define a recurrence equation for the computation of the minimum cost for arriving to the node  $(k, t)$ :

$$\begin{aligned}
 C_{(0,0)} &= 0 \\
 C_{(k,t)} &= \min_{(k',t') \in A_{k,t}} \{C_{(k',t')} + V_{(k',t')-(k,t)}\}
 \end{aligned}
 \tag{10}$$

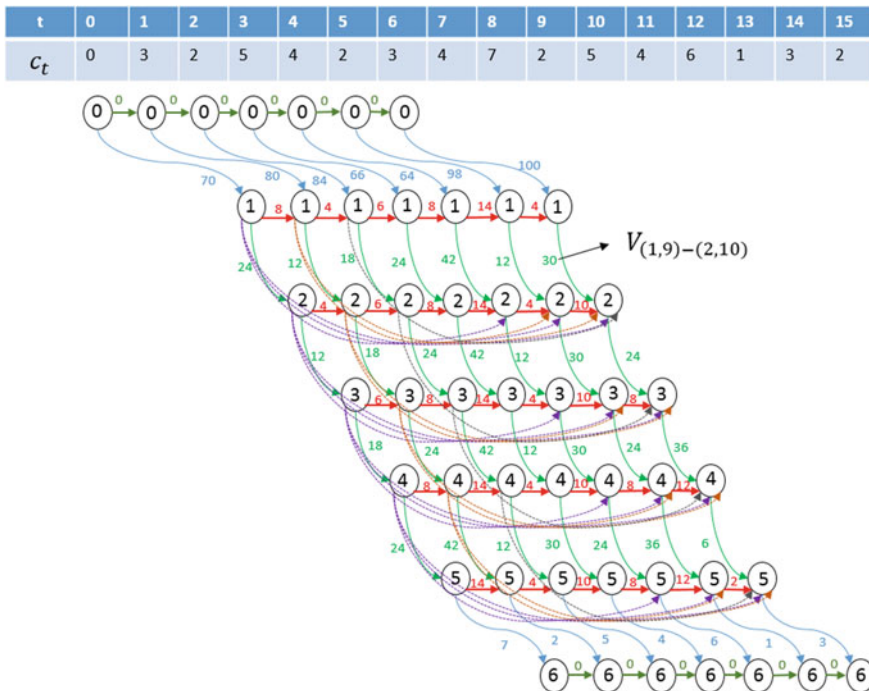


Fig. 2 Graph representation for a instance with 5 process times and 15 periods

**Table 1** The obtained results for CPLEX (1) and Dijkstra (2) methods

(P, T)	Obj <sub>1</sub>	CPU <sub>1</sub>	Obj <sub>2</sub>	CPU <sub>2</sub>	(P,T)	Obj <sub>1</sub>	CPU <sub>1</sub>	Obj <sub>2</sub>	CPU <sub>2</sub>	(P,T)	Obj <sub>1</sub>	CPU <sub>1</sub>	Obj <sub>2</sub>	CPU <sub>2</sub>
(3, 10)	108	1.96	108	0	(74, 92)	1667	1.59	1667	0.03	(177, 249)	3850	4.74	3850	2.12
(5, 20)	137	0.74	137	0	(86, 107)	1871	3.72	1871	0.05	(200, 270)	4456	16.86	4456	2.66
(16, 23)	322	0.21	322	0	(102, 126)	2204	1.55	2204	0.06	(213, 300)	4759	24.42	4759	4.39
(25, 40)	522	1.26	522	0	(119, 134)	2565	5.13	2565	0.03	(246, 360)	5366	12.7	5366	4.60
(37, 48)	789	0.77	789	0	(136, 153)	2989	2.89	2989	0.06	(280, 420)	5881	17.53	5881	10.04
(46, 59)	1032	0.88	1032	0	(153, 172)	3443	6.16	3443	0.09	(328, 480)	6765	33.83	6765	18.91
(57, 72)	1248	1.15	1248	0	(156, 209)	3433	5.16	3433	1.04					

where,  $A_{k,t}$  is set of the precedent nodes that are connected to node  $(k, t)$  directly. For example, in Fig. 2,  $A_{4,11} = \{(4, 10), (3, 10), (3, 6), (3, 5)\}$ . Finally, the amount of  $C_{(P+1,T)}$  represents the value of objective function for our problem.

In the previous sections of this paper, at first the considered problem is modeled by a dynamic programming method, then Dijkstra's algorithm is used to find the optimal solution. Therefore, the complexity of this problem is equal to complexity of the Dijkstra's algorithm for this problem.

According to [5], the implementation of Dijkstra's algorithm based on a min-priority queue, runs in  $O(|E| + |V| \log |V|)$  (where  $|E|$  is the number of edges and  $|V|$  is the number of vertices or nodes). Consequently, the complexity of this algorithm for the presented problem is equal to:

$$O(T^2P + TP \log TP) = O(T^2P + TP \log T + TP \log P) \cong O(T^2P) \quad (11)$$

Since, the largest possible value of  $P$  is certainly less than  $T$  (worst case analysis), so, the optimal solution of this problem may be obtained in polynomial time ( $O(T^3)$ ).

## 5 Numerical Experiments

Some numerical experiments are presented to show the effectiveness of the proposed approach. Also, the results of this approach is compared with the linear programming method which is implemented on ILOG CPLEX Software. During this study, a computer with 2.6 GHz Intel Core i5 processor and 8 GB of RAM was used to perform all the experiments. Table 1 represents a part of the obtained results for the problem with  $P$  process times and  $T$  periods. The results illustrate the effectiveness of Dijkstra's algorithm in all the cases with less time-consuming than CPLEX.

## 6 Conclusion

In this paper, the complexity of the preemption case of a single machine scheduling problem with state-dependent and time-dependent energy cost is investigated when the objective is the total energy consumption costs minimization. This problem is modelled with a finite graph and then, the optimal schedule for the machine state in each period is found by the Dijkstra's algorithm during a polynomial time. The complexity analysis of this algorithm proved that, the considered problem is a polynomial problem of degree 3 ( $O(T^3)$ ). Finally, the presented method is applied for several instances. The obtained results by CPLEX software (Branch and Cut method) and Dijkstra's algorithm (Dynamic programming method) for the same instances, showed that in all the cases the proposed method finds the optimal solution in a few seconds.

As future study, an extension of the presented dynamic programming approach, for the non-preemption case of this problem, as well as, scheduling problems of a single machine with more assumptions and constraints are envisioned.

## References

1. Aghelinejad, M., Ouazene, Y., Yalaoui, A.: Machine and production scheduling under electricity time varying prices. In: 2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), pp. 992–996. IEEE (2016)
2. Antoniadis, A., Huang, C.-C., Ott, S.: A fully polynomial-time approximation scheme for speed scaling with sleep state. In: Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1102–1113. Society for Industrial and Applied Mathematics (2015)
3. Che, A., Zeng, Y., Lyu, K.: An efficient greedy insertion heuristic for energy-conscious single machine scheduling problem under time-of-use electricity tariffs. *J. Clean. Prod.* (2016)
4. Fang, K., Uhan, N.A., Zhao, F., Sutherland, J.W.: Scheduling on a single machine under time-of-use electricity tariffs. *Ann. Oper. Res.* 1–29 (2014)
5. Fredman, M.L., Tarjan, R.E.: Fibonacci heaps and their uses in improved network optimization algorithms. *J. ACM (JACM)* **34**(3), 596–615 (1987)
6. Gong, X., De Pessemer, T., Joseph, W., Martens, L.: An energy-cost-aware scheduling methodology for sustainable manufacturing. *Procedia CIRP* **29**, 185–190 (2015)
7. Gong, X., De Pessemer, T., Joseph, W., Martens, L.: A generic method for energy-efficient and energy-cost-effective production at the unit process level. *J. Clean. Prod.* **113**, 508–522 (2016a)
8. Gong, X., De Pessemer, T., Joseph, W., Martens, L.: A power data driven energy-cost-aware production scheduling method for sustainable manufacturing at the unit process level. In: 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA), pp. 1–8. IEEE (2016b)
9. Masmoudi, O., Yalaoui, A., Ouazene, Y., Chehade, H.: Solving a capacitated flow-shop problem with minimizing total energy costs. *Int. J. Adv. Manuf. Technol.* 1–13 (2016). <https://doi.org/10.1007/s00170-016-9557-5>
10. Masmoudi, O., Yalaoui, A., Ouazene, Y., Chehade, H.: Lot-sizing in a multi-stage flow line production system with energy consideration. *Int. J. Prod. Res.* **55**(6), 1640–1663 (2017). <https://doi.org/10.1080/00207543.2016.1206670>
11. Mikhaylidi, Y., Naseraldin, H., Yedidsion, L.: Operations scheduling under electricity time-varying prices. *Int. J. Prod. Res.* **53**(23), 7136–7157 (2015)
12. Mouzon, G., Yildirim, M.B.: A framework to minimise total energy consumption and total tardiness on a single machine. *Int. J. Sustain. Eng.* **1**(2), 105–116 (2008)
13. Mouzon, G., Yildirim, M.B., Twomey, J.: Operational methods for minimization of energy consumption of manufacturing equipment. *Int. J. Prod. Res.* **45**(18–19), 4247–4271 (2007)
14. Shrouf, F., Ordieres-Meré, J., García-Sánchez, A., Ortega-Mier, M.: Optimizing the production scheduling of a single machine to minimize total energy consumption costs. *J. Clean. Prod.* **67**, 197–207 (2014)
15. Yildirim, M.B., Mouzon, G.: Single-machine sustainable production planning to minimize total energy consumption and total completion time using a multiple objective genetic algorithm. *IEEE Trans. Eng. Manag.* **59**(4), 585–597 (2012)



# An Optimization Model for the Outbound Truck Scheduling Problem at Cross-Docking Platforms

Antonio Diglio, Andrea Genovese and Carmela Piccolo

**Abstract** A cross-dock is a facility where arriving materials are sorted, grouped and delivered to destinations, with very limited storage times, with the overall objective of optimizing the total management costs. The operational efficiency of a cross-docking system strongly depends on how the logistic activities are organized. For this reason, optimization models and methods can be very useful to improve the system performances. In this paper, we propose a mathematical model to describe the so-called truck scheduling problem at a cross-docking platform. The model considers most of the actual constraints occurring in real problems; therefore, it can be viewed as an interesting basis to define a decision support system for this kind of problems. Some preliminary results show that the model can be efficiently solved in limited computational times.

**Keywords** Supply chain · Cross-docking · Truck scheduling

## 1 Introduction

A cross-dock is a facility that receives goods from suppliers and sorts them into alternative groups, based on the downstream delivery points. This way, it is possible to reduce the total distribution costs taking advantage of the benefits of a warehousing strategy in terms of consolidation (enabling economies of scale

---

A. Diglio · C. Piccolo (✉)

Department of Industrial Engineering, University of Naples Federico II,  
Piazzale Tecchio 80, 80125 Naples, Italy  
e-mail: carmela.piccolo@unina.it

A. Diglio  
e-mail: antonio.diglio@unina.it

A. Genovese  
Management School, University of Sheffield, Conduit Road,  
S10 1FL Sheffield, Sheffield, UK  
e-mail: a.genovese@sheffield.ac.uk

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_60

through the consolidation of multiple less-than-truckload shipments), but avoiding storage costs and reducing handling operations.

In the literature, several models and methods have been proposed to address different kinds of cross-docking optimization problems. In this paper, we propose a mathematical model to describe the so-called truck scheduling problem at a cross-dock. The model considers most of the actual constraints occurring in real-world problems, striving to define a decision support system for the management of cross-docking platforms.

The remainder of the paper is arranged as follows. In the next section, a review of the extant literature dealing with cross-docking optimization highlights the main research strands and potential gaps. Then, the description of the model is provided. Computational results are then illustrated, showing that the model can be efficiently solved in limited computational times.

## 2 Literature Review

In recent years, many attempts have been made in order to systematize the cross-docking literature [1–5]. In particular, Buijs et al. [5] classified problems on the basis of spatial and temporal aspects, distinguishing between single cross-dock and cross-docking network management problems, and, according to the temporal dimension, among strategic, medium term and operational problems. Most of the literature concerns with operational decision-making problems at a local level; in particular with:

- the *inbound truck scheduling problem*, consisting in the assignment of the inbound trucks to the receiving doors and of their subsequent scheduling;
- the *outbound truck scheduling problem*, arising when, starting from the inbound trucks' arrival scheduling, the loading and the scheduling of outbound vehicles should be determined, along with their assignment to shipping doors;
- the *synchronization truck scheduling problem*, arising when both the previous problems have to be simultaneously solved.

Considering the interdependencies among the above problems, global optimization approaches should include all the planning and management aspects. However, the relevant complexity of the single sub-problems suggests the development of separated models and methods.

Most of the papers in the literature consider very simplifying assumptions, representing the cross-docking facility with one receiving door, one shipping door and an infinite staging area capacity [6–10]. With a slight variation, Vahdani and Zandieh [11] and Soltani and Saldjadi [12] considered a cross-dock that does not allow storage. Chen and Lee [13] solved the one inbound—one outbound door truck scheduling, modelling it as a detailed scheduling problem. Alpan et al. [14] dealt with a multi-door cross docking problem, considering temporary and limited storage. Boysen et al. [15, 16] determined the schedule sequences for inbound

trailers in a multiple door cross-dock. Similar problems were also tackled by Liao et al. [17], who also dealt with inbound truck assignment to dock doors and outbound trucks. Konur and Golias [18] introduced uncertainty in inbound truck arrival times, assuming just arrival time windows are known.

Miao et al. [19] were the first to consider a synchronization problem in a multiple doors cross-dock. Chen and Song [20] extended the work of Chen and Lee [13], to solve the problem with multiple doors for inbound and outbound processes. The integration of Vehicle Routing Problem into a cross-docking system was also considered [21–23].

Most of the proposed cross-docking scheduling problems deal with fixed outbound schedules and just optimise inbound truck processing. Indeed, Boysen et al. [16] mention that fixed outbound schedule represents an important real-world aspect of cross docking. This is true especially in large hub-and-spoke networks. However, in industries characterised by less-than-truckload logistics and specific deadlines for goods (such as postal services or food supply chains), where firms mainly transport comparatively small and low-valued shipments of multiple senders, things may change quite significantly. Indeed, in this kind of industry the need for consolidation, load optimization and minimization of number of outbound trucks, while respecting deadlines, is even more important, due to tight profit margins. For this reason, this paper will propose a mathematical programming framework to deal with outbound truck scheduling at cross-docking platforms. The model assumes that inbound truck sequencing is known a priori, and seeks to minimize the number of departures towards a set of destinations. The proposal may be viewed as an extension of models introduced by Bruno et al. [24, 25] for the bus scheduling at a transit terminal. In the next section, the model will be described firstly considering the basic case of one inbound—one outbound door, and then extended to the general case with multiple inbound and outbound doors.

### 3 A General Framework Model for Cross-Docking Truck Scheduling

#### 3.1 *One Inbound—One Outbound Door Case*

For sake of clarity, we initially describe the proposed model with reference to the outbound truck scheduling problem for the one inbound—one outbound door case (see also Bruno et al. [26]).

In particular, dividing the time horizon into  $n$  time periods, the time expansion of the terminal over the horizon  $[0, T]$ , is given by a graph of  $n$  nodes, in which each node corresponds to a copy of the terminal in the time  $t$ , that is linked with the node  $t + 1$  by an holdover arc (Fig. 1a).

At a given time, an inbound truck delivers a certain number of lots at the door of the cross-dock. Each lot is generally characterized by a destination  $o$  and a deadline  $d$ , within which it needs to leave the dock. Sets of lots with the same destination can

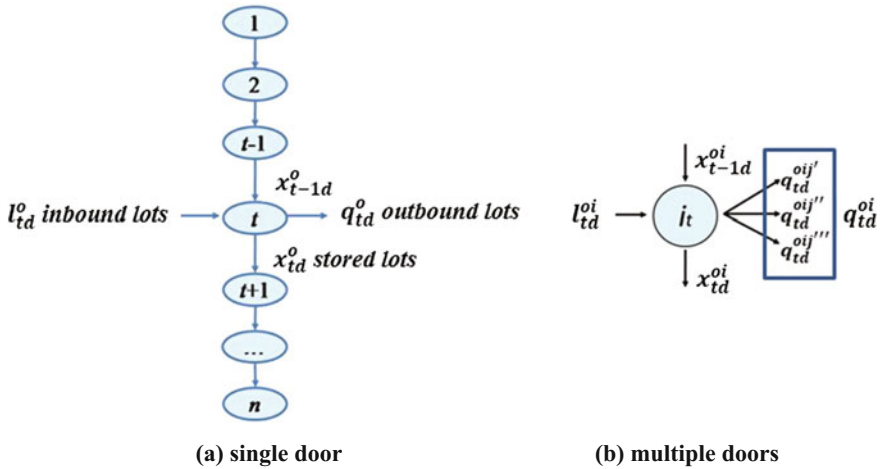


Fig. 1 Cross docking dynamic network

be grouped together and leave the cross-dock in  $t$ , if there is an outbound truck toward that destination in  $t$ . Otherwise, they can be stored in the terminal, flowing through the holdover arcs.

Assuming as parameters:

- $O$ , set of the possible destinations ( $o \in O$ );
- $D$ , set of the possible deadlines ( $d \in D$ );
- $l_{td}^o$ . Lots with deadline  $d$  and destination  $o$  arrived at the cross-dock at time  $t$ ;
- $f_t$ , cost associated with the vehicle leaving the cross-dock at time  $t$ ;
- $Q_t$ , capacity of the outbound vehicles, i.e. the maximum number of lots which can be loaded on an outbound truck in period  $t$ ;
- $C$ , capacity of the cross-dock, i.e. the maximum number of lots that can be stored at the facility (assumed to be constant over the time);

and introducing as decision variables:

- $x_{td}^o$ , number of lots with deadline  $d$  and destination  $o$  stored in  $[t, t + 1]$ ;
- $q_{td}^o$ , number of lots with deadline  $d$  and destination  $o$  leaving at time  $t$ ;
- $y_t^o$ , binary variable equal to 1 if a truck leaves the cross-dock at time  $t$  toward the destination  $o$ , 0 otherwise;

the formulation of the model is given by:

$$\min z = \sum_{t=1}^n \sum_{o \in O} f_t y_t^o \tag{1}$$

$$x_{td}^o = x_{t-1,d}^o + l_{td}^o - q_{td}^o \quad \forall t = 1, \dots, n; \forall o \in O; \forall d \in D \tag{2}$$

$$\sum_{d \geq t} q_{td}^o \leq Q_t y_t^o \quad \forall t = 1, \dots, n; \quad \forall o \in O \tag{3}$$

$$\sum_{o \in O} \sum_{d \in D} x_{td}^o \leq C \quad \forall t = 1, \dots, n \tag{4}$$

$$\sum_{o \in O} y_t^o \leq 1 \quad \forall t = 1, \dots, n \tag{5}$$

$$x_{0d}^o = x_{dd}^o = 0 \quad \forall o \in O; \forall d \in D \tag{6}$$

$$x_{td}^o, q_{td}^o \geq 0 \quad \forall t = 1, \dots, n; \forall o \in O; \forall d \in D \tag{7}$$

$$y_t^o \in \{0; 1\} \quad \forall t = 1, \dots, n; \forall o \in O. \tag{8}$$

The objective function (1) is the sum of the costs associated with the activation of outbound trucks across the planning horizon. Constraints (2) are the so-called mass balance constraints, i.e. the conditions about the flow material balance at each time  $t$ . Conditions (3) and (4) assure that capacity constraints of the cross-dock and of the outbound trucks are satisfied in each time period. Constraints (5) indicate that no more than one truck can leave the cross-dock in each time  $t$ , because of the presence of a single shipping door; constraints (6) are related to the deadlines as they assure that no lot stay inside the cross-dock after their own deadlines. Constraints (7–8) define the nature of the introduced variables.

### 3.2 The Multiple Doors Case

In this case, more than one truck can arrive at and depart from the terminal at each time  $t$  from different doors. Then, two further indices have been introduced to identify the receiving door  $i$  and the shipping door  $j$  respectively. The lots arriving at a given receiving door  $i$  ( $l_{td}^o$ ) can be temporary stored or directly moved towards one of the shipping doors  $j$  for departure. Lots are stored at the related receiving doors and picked up whenever they have to be loaded on a truck at a given door  $j$ . The times to transfer lots from receiving to shipping doors cannot be neglected; in particular, they have been assumed dependent on the specific pair  $(i, j)$ .

Considering the following further notation:

- $I$ , set of the inbound doors ( $i \in I$ );
- $J$ , set of the outbound doors ( $j \in J$ );
- $m_{ij}$ , time to transfer lots from the receiving door  $i$  to the shipping door  $j$ ;

and introducing as decision variables (Fig. 1b):

- $x_{td}^{oi}$ , number of lots coming from the receiving door  $i$ , with deadline  $d$  and destination  $o$  stored during the time interval  $[t, t + 1]$ ;
- $q_{td}^{oj}$ , number of lots coming from the receiving door  $i$  with deadline  $d$  and destination  $o$  leaving at time  $t$  from the shipping doors  $j$ ;
- $y_t^{oj}$ , binary variable equal to 1 if a truck leaves from the shipping door  $j$  at time  $t$  to the destination  $o$ , 0 otherwise.

the model can be formulated as follows:

$$\min z = \sum_{j \in J} \sum_{t=1}^n \sum_{o \in O} f_t y_t^{oj} \quad (9)$$

$$x_{td}^{oi} = x_{(t-1)d}^{oi} + l_{td}^{oi} - \sum_{j \in J: (t+m_{ij}) \leq n} q_{(t+m_{ij})d}^{oj} \quad (10)$$

$$\forall t = 1, \dots, n; \forall o \in O; \forall i \in I; \forall d \in D$$

$$\sum_{o \in O} \sum_{i \in I} \sum_{d \in D} x_{td}^{oi} \leq C \quad \forall t = 1, \dots, n \quad (11)$$

$$\sum_{d \in D} q_{td}^{oj} \leq Q_t y_t^{oj} \quad \forall t = 1, \dots, n; \forall o \in O; \forall j \in J \quad (12)$$

$$\sum_{o \in O} y_t^{oj} \leq 1 \quad \forall t = 1, \dots, n; \forall j \in J \quad (13)$$

$$x_{0d}^{oi} = 0 \quad \forall o \in O; \forall i \in I; \forall d \in D \quad (14)$$

$$\sum_{j \in J} \left[ \sum_{t=1}^{d-m_{ij}} q_{td}^{oj} \right] = \sum_{t=1}^d l_{td}^{oi} \quad \forall o \in O; \forall i \in I; \forall d \in D \quad (15)$$

$$x_{td}^{oi} \geq 0 \quad \forall t = 1, \dots, n; \forall o \in O; \forall i \in I; \forall d \in D \quad (16)$$

$$q_{td}^{oj} \geq 0 \quad \forall t = 1, \dots, n; \forall o \in O; \forall i \in I; \forall d \in D \quad (17)$$

$$y_t^{oj} \in \{0; 1\} \quad \forall t = 1, \dots, n; o \in O; j \in J \quad (18)$$

The objective function (9) is defined as the sum of the costs associated with the departure of outbound trucks during the planning horizon. Constraints (10) indicate the flow material balance at each receiving door  $i$  at each  $t$ . Conditions (11) assure

that, in each period  $t$ , the lots stored at all the receiving doors  $i$  do not exceed the total capacity  $C$  of the dock. Conditions (12) guarantee that if in  $t$  a truck leaves from a generic shipping door  $j$  toward the destination  $o$  ( $y_t^{oj} = 1$ ), only the lots with the same destination can be loaded on the truck, without exceeding its capacity. Constraints (13) indicate that no more than one truck can depart from the shipping door  $j$  at each time  $t$ . Constraints (14) impose that the stock level at each receiving door  $i$  is zero at the beginning of the planning horizon; while (15) assure that no lot stays inside the cross-dock after its own deadline. Conditions (16–18) define the nature of the introduced decision variables. It has to be noticed that conditions (15) can be also alternatively formulated as follows:

$$q_{td}^{oj} = 0 \quad \forall i \in I; \forall o \in O; \forall j \in J; \forall d \in D; \forall t = (d + 1), \dots, n \quad (19)$$

$$x_{td}^{oi} = 0 \quad \forall i \in I; \forall o \in O; \forall d \in D; \forall t = \left( d - \min_{j \in J} m_{ij} + 1 \right), \dots, n \quad (20)$$

Constraints (19) impose that lots with deadline  $d$  cannot be loaded on a truck later than  $d$ , at any shipping door  $j$ . Conditions (20) indicate the last period in which lots with deadline  $d$  may be stored at each receiving door  $i$ . In particular, for each  $i$ , there exists a latest period in which lots with deadline  $d$  can be picked up and moved toward the shipping door  $j$  ( $t_{ij}^d = d - m_{ij}$ ). After that, they may be transferred only towards those doors  $j$  reachable from  $i$  in less than  $m_{ij}$ . Then, the latest period in which lots with deadline  $d$  may be picked up from  $i$  is  $t_i^d = d - \min_{j \in J} m_{ij}$ .

## 4 Computational Experiences

In order to test the suitability of the proposed model, a set of instances was produced using a random generator designed and implemented in C++ language. ILOG CPLEX Optimization Tool was used to solve the randomly generated instances for the case of unlimited dock capacity, by varying the number of time periods  $T$  (from 12 to 36), the number of destinations  $O$  (from 2 to 6) and the number of inbound and outbound doors  $|I| = |J|$  (from 2 to 6). Results, reported in Table 1, show that computational times grow in a reasonable way in all cases. Even in the case of 36 times periods, 6 destinations and 12 doors (6 inbound and 6 outbound doors,) the solver is capable of finding a solution to the problem within four minutes.

**Table 1** Run Times (s)

<i>T = 12</i>									
III = IJI	OI = 2			OI = 4			OI = 6		
	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
2	0.30	0.18	0.38	0.47	0.38	0.53	0.63	0.35	0.91
4	0.31	0.29	0.32	1.50	0.66	2.10	18.00	1.70	75.00
6	0.64	0.28	1.46	1.53	0.64	2.42	32.00	1.30	145.00
<i>T = 18</i>									
III = IJI	OI = 2			OI = 4			OI = 6		
	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
2	0.34	0.25	0.45	0.64	0.39	0.98	0.71	0.39	0.98
4	0.61	0.37	0.84	9.10	0.70	24.00	28.00	21.00	40.00
6	1.80	0.62	3.26	10.30	3.90	27.00	146.00	68.00	214.00
<i>T = 24</i>									
III = IJI	OI = 2			OI = 4			OI = 6		
	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
2	0.70	0.37	1.28	0.82	0.57	1.32	1.20	0.79	1.61
4	0.79	0.58	0.94	9.50	1.90	31.00	52.00	7.50	121.00
6	2.40	1.00	3.40	51.00	22.00	82.00	158.00	71.00	220.00
<i>T = 36</i>									
III = IJI	OI = 2			OI = 4			OI = 6		
	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max
2	1.70	0.64	3.09	2.60	1.20	3.60	3.40	1.30	5.70
4	1.90	1.00	3.10	16.40	4.30	18.70	56.00	13.00	58.00
6	11.30	6.50	15.30	66.00	18.70	73.00	241.00	93.00	265.00

## 5 Conclusions

In this paper, we analyzed the cross-docking approach as a tool to improve the performance of a delivery system within a supply chain context. Literature on this topic has shown that efficiency and effectiveness of cross-docking strategies strongly depend on the availability of optimization models and algorithms able to support the decision maker about the choices on the operational aspects. However, current literature on the truck scheduling just provides models in order to solve specific versions of the problem. For this reason, a general framework has been introduced, based on a mathematical model able to describe most of the scenarios, which can occur in practical applications. The first provided results show that the model produces interesting results both in term of computational efficiency and from a managerial point of view. Further investigations will be aimed at improving the computational efficiency of the model, through purpose-built solution methodologies.



**Acknowledgements** This research was partially supported by the project “Promoting Sustainable Freight Transport in Urban Contexts: Policy and Decision-Making Approaches (ProSFFeT)”, funded by the H2020-MSCA-RISE-2016 programme (Grant Number: 734909).

## References

1. Agustina, D., Lee, C.K.M., Piplani, R.: A review: mathematical models for cross-docking planning. *Int. J. Eng. Bus. Manage.* **2**(2), 47–54 (2010)
2. Boysen, N., Flidner, M.: Cross-dock scheduling: Classification, literature review and research agenda. *Omega* **38**, 413–422 (2010)
3. Stephan, K., Boysen, N.: Cross-docking. *J. Manag. Control* **22**, 129–137 (2011)
4. Van Belle, J., Valckenaers, P., Cattrysse, D.: Cross-docking: state of the art. *Omega* **40**, 827–846 (2012)
5. Buijs, P., Vis, I.F., Carlo, H.J.: Synchronization in cross-docking networks: a research classification and framework. *Eur. J. Oper. Res.* **239**, 593–608 (2014)
6. Yu, W., Egbelu, P.J.: Scheduling of inbound and outbound trucks in cross-docking systems with temporary storage. *Eur. J. Oper. Res.* **184**, 377–396 (2008)
7. Arabani, A.B., Ghomi, S.F., Zandieh, M.: A multi criteria cross-docking scheduling with just-in-time approach. *The International Journal of Advancing Manufacturing Technology* **49** (5–8), 741–756 (2010)
8. Boysen, N.: Truck scheduling at zero-inventory cross-docking terminals. *Comput. Oper. Res.* **37**, 32–41 (2010)
9. Forouharfard, S., Zandieh, M.: An imperialist competitive algorithm to schedule of receiving and shipping trucks in a cross-docking system. *The International Journal of Advancing Manufacturing Technology* **51**(9), 1179–1193 (2010)
10. Larbi, R., Alpan, G., Baptiste, P., Penz, B.: Scheduling cross docking operations under full, partial and no information on inbound arrivals. *Comput. Oper. Res.* **38**(6), 889–90 (2011)
11. Vahdani, B., Zandieh, M.: Scheduling trucks in a cross-docking system: robust meta-heuristics. *Comput. Ind. Eng.* **58**(1), 12–24 (2010)
12. Soltani, R., Saldjadi, S.J.: Scheduling trucks in a cross-docking system: a robust meta-heuristic approach. *Transp. Res. Part E* **46**(5), 650–666 (2010)
13. Chen, F., Lee, C.-Y.: Minimizing the makespan in a two-machine cross-docking flow shop problem. *Eur. J. Oper. Res.* **193**, 59–72 (2009)
14. Alpan, G., Larbi, R., Penz, B.: A bounded dynamic programming approach to schedule operations in a cross docking platform. *Comput. Ind. Eng.* **60**, 385–396 (2011)
15. Boysen, N., Flidner, M., Scholl, A.: Scheduling inbound and outbound trucks at cross-docking terminals. *OR Spectrum* **32**, 135–161 (2010)
16. Boysen, N., Briskorn, D., Tschöke, M.: Truck scheduling in cross-docking terminals with foxed outbound departure. *OR Spectrum* **35**, 479–504 (2013)
17. Liao, T.W., Egbelu, P.J., Chang, P.C.: Simultaneous dock assignment and sequencing of inbound trucks under a fixed outbound truck schedule in multi-door cross docking operations. *Int. J. Prod. Econ.* **141**, 212–229 (2013)
18. Konur, D., Goliass, M.M.: Cost-stable truck scheduling at a cross-dock facility with unknown truck arrivals: a meta-heuristic approach. *Transp. Res. Part E* **49**, 71–91 (2013)
19. Miao, Z., Lim, A., Ma, H.: Truck dock assignment with operational time constraint within crossdocks. *Eur. J. Oper. Res.* **192**, 105–115 (2009)
20. Chen, F., Song, K.: Minimizing makespan in a two-stage hybrid cross-docking scheduling problem. *Comput. Oper. Res.* **36**, 2066–2073 (2009)
21. Liao, C.J., Lin, Y., Shih, S.C.: Vehicle routing with cross-docking in the supply chain. *Expert Syst. Appl.* **37**(10), 6868–6873 (2010)

22. Santos, F.A., Mateus, G.R., da Cunha, A.S.: The pickup and delivery problem with cross-docking. *Comput. Oper. Res.* **40**(4), 1085–1093 (2013)
23. Grangier, P., Gendreau, M., Lehuédé, F., Rousseau, L.-M.: A matheuristic based on large neighborhood search for the vehicle routing problem with cross-docking. *Comput. Oper. Res.* **84**, 116–126 (2017)
24. Bruno, G., Genovese, A., Sgalambro, A.: An extension of the schedule optimization problem at a public transit terminal to the multiple destinations case. *Public Transport* **3**(3), 189–198 (2011)
25. Bruno, G., Improta, G., Sgalambro, A.: Models for the schedule optimization problem at a public transit terminal. *OR Spectrum* **31**(3), 465–481 (2009)
26. Bruno, G., Genovese, A., Piccolo, C.: The capacitated Lot Sizing model: A powerful tool for logistics decision making. *Int. J. Prod. Econ.* **155**, 380–390 (2014)

# Min-Max Regret Scheduling to Minimize the Total Weight of Late Jobs with Interval Uncertainty

Maciej Drwal

**Abstract** We study the single machine scheduling problem with the objective to minimize the total weight of late jobs. It is assumed that the processing times of jobs are not exactly known at the time when a complete schedule must be dispatched. Instead, only interval bounds for these parameters are given. In contrast to the stochastic optimization approach, we consider the problem of finding a robust schedule, which minimizes the maximum regret of a solution. Heuristic algorithm based on mixed-integer linear programming is presented and examined through computational experiments.

**Keywords** Robust optimization • Mixed integer programming • Uncertainty

## 1 Introduction

We consider the following fundamental scheduling problem. A set of jobs is given to be processed on a single machine. Each job requires possibly different processing time to complete and cannot be interrupted or preempted. There is a fixed due-date until which the work should be finished. However, it is uncertain how much processing each of the task would exactly take. Before the schedule is dispatched on the machine, the only available data is the set of interval bounds, to which the actual processing requirements belong. The goal is to sequence the jobs, so that the number of the jobs that complete before the due-date is maximal (or, equivalently, the number of late jobs is minimal). In a more general problem variant, each job is associated with a weight (or cost), and the objective is to minimize the sum of weights of late jobs.

This problem arises in many diverse application areas. For instance, this situation is experienced by a client who leases a fixed machine time (e.g., in a computing

---

M. Drwal (✉)

Department of Computer Science, Wrocław University of Science and Technology,  
Wybrzeże Wyspińskiego 27, 53-370 Wrocław, Poland  
e-mail: maciej.drwal@pwr.edu.pl

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_61

center) to carry out a number of tasks, but each of them requires unknown processing time; on the other hand, upper bounds on the processing times are set. This problem may also occur in a manufacturing process, when a fixed due-date is set for a batch of finished items to be delivered, but production time of each item may vary within known bounds.

The processing times uncertainty can be handled in a several different ways. One common approach is to use stochastic framework, and model the quantities of interest as random variables. This has its advantages in specific situations; however, it often brings the need for collecting data in order to estimate parameters. Moreover, in certain critical applications, the probabilistic guarantees, offered by such an approach, may not be sufficient. In this paper, we consider a *robust optimization* approach [3, 7]. Each realization of uncertain parameters is treated as equally possible. Our aim is to come up with such a solution that degrades the least as compared to the best solution in every possible scenario. This measure of solution quality is reflected in the notion of *maximum regret* [9].

Most of the basic scheduling problems have been already considered within the robust optimization framework [1, 6, 8]. The majority of these works concerns the more restrictive case of discrete uncertainty (finitely many possible realizations of parameters). If the processing times were known precisely, the unweighted variant of the problem considered in this paper could be solved in polynomial time [4]. However, even for 2 processing times scenarios, it becomes NP-hard [2]. The case of interval processing times, described in the next section, appears to occur more naturally in practice. Although the number of processing times scenarios in such case is potentially infinite, solution algorithms may take the advantage of the structural information of uncertainty sets. Unfortunately, the problem with interval data is also NP-hard [5], even if all weights are equal. Moreover, deterministic variant with arbitrary weights is already NP-hard. A viable solution approach is the application of mathematical programming techniques, presented in this paper.

## 2 Problem Formulation

The deterministic version of the considered scheduling problem is denoted  $1|d_i = d|\sum w_i U_i$ . Given is the set of jobs  $J = \{1, 2, \dots, n\}$ . Each job  $j \in J$  is described by the processing time  $p_j$  and weight  $w_j$ . Let  $d > 0$  denote the due-date. A solution (schedule) is a permutation  $\pi = (\pi(1), \pi(2), \dots, \pi(n))$ , where  $\pi(k)$  is the index of job scheduled to be executed as  $k$ th from the start. Equivalently, we encode the solution as a binary matrix  $\mathbf{x}$ , where  $x_{kj} = 1$ , iff  $j$ th job is scheduled on position  $k$  from the start, and  $x_{kj} = 0$  otherwise. The completion time of job scheduled on position  $k$  is defined as:

$$C(\mathbf{x}, k) = \sum_{i=1}^k \sum_{j \in J} x_{ij} p_j.$$

We define  $U(\mathbf{x}, k) = 0$ , iff  $C(\mathbf{x}, k) \leq d$ ; we say that the job on position  $k$  is *on-time*. Otherwise,  $U(\mathbf{x}, k) = \sum_{j \in J} x_{kj} w_j$ , and we say that the job on position  $k$  is *late*. An optimal schedule is one that minimizes the weighted number of late jobs,  $F(\mathbf{x}) = \sum_{k=1}^n U(\mathbf{x}, k)$ . An important special case, when  $w_j = 1$  for all  $j \in J$ , is the problem of minimizing only the number of late jobs.

In an uncertain problem, for each  $j \in J$ , instead of exact processing times  $p_j$ , we are given interval bounds  $p_j^-, p_j^+$ , so that the actual processing time can be any real number between them. A vector of processing times will be called a *scenario*. The set of all possible scenarios is defined as:

$$\mathcal{U} = \{\mathbf{p} = (p_1, \dots, p_n) : \forall_{j \in J} p_j^- \leq p_j \leq p_j^+\}.$$

The value of objective function in a scenario  $\mathbf{p} \in \mathcal{U}$  will be denoted by  $F(\mathbf{x}, \mathbf{p})$ .

Let  $\mathcal{P}$  be the set of all  $n$ -by- $n$  permutation matrices. Given a solution  $\mathbf{x} \in \mathcal{P}$ , and a scenario  $\mathbf{p} \in \mathcal{U}$ , we define the regret as:

$$R(\mathbf{x}, \mathbf{p}) = F(\mathbf{x}, \mathbf{p}) - \min_{\mathbf{y} \in \mathcal{P}} F(\mathbf{y}, \mathbf{p}).$$

A schedule represented by matrix  $\mathbf{y}$  in this context will be called an *adversarial* schedule. Then the maximum regret is denoted as:

$$Z(\mathbf{x}) = \max_{\mathbf{p} \in \mathcal{U}} R(\mathbf{x}, \mathbf{p}). \quad (1)$$

We will also use the notation  $Z(\pi)$  to denote the maximum regret  $Z(\mathbf{x})$  of a matrix  $\mathbf{x}$  equivalent to permutation  $\pi$ .

A scenario that maximizes the regret will be called a *worst-case scenario*. A *robust optimal* solution  $\mathbf{x}^*$  is one that minimizes the maximum regret:

$$Z^* = Z(\mathbf{x}^*) = \min_{\mathbf{x} \in \mathcal{P}} Z(\mathbf{x}). \quad (2)$$

### 3 Computation of Maximum Regret

An essential prerequisite for solving the robust problem (2) is the solution for the subproblem of regret maximization (1). Let us fix a schedule  $\pi$ . Since a due-date  $d$  is common for all jobs, there exists a job on such a position  $l$  in  $\pi$ , so that all jobs  $\pi(1), \pi(2), \dots, \pi(l-1)$ , are on-time, while all jobs  $\pi(l), \pi(l+1), \dots, \pi(n)$ , are late. Observe that worst-case scenario for  $\pi$  is one for which the difference between the total weight of late jobs in  $\pi$ , and the total weight of late jobs in adversarial schedule is maximal. In the special case of equal weights, each late job contributes equally to the value of objective function, thus for any fixed scenario, an adversarial schedule is constructed by sorting all jobs with respect to nondecreasing processing times. This

is not true for the case of general weights, where computing adversarial schedule for a fixed scenario is equivalent to solving an instance of knapsack problem.

Intuitively, in the worst-case schedules, the jobs that complete before the due-date  $d$  would have the processing time closer to their respective upper bounds of uncertainty intervals. On the other hand, late jobs would generally have shorter worst-case processing times, closer to their lower bounds of uncertainty intervals. Such processing times allow for the late jobs to be early in the adversarial schedule, maximizing the number of on-time jobs.

Let us consider the following example problem instance with  $n = 3$  identical jobs. Each has the same processing time interval  $[p_j^-, p_j^+] = [1, 3]$ , for  $j \in \{1, 2, 3\}$ . Let the due-date be equal to 5. Since the jobs are identical, the maximum regret is the same for each schedule, thus let  $\pi = (1, 2, 3)$ . It can be seen that the following processing times constitute a worst-case scenario:  $p_1 = 3, p_2 = 2 + a$ , for  $a \in (0, 1]$ , and  $p_3 = 1$ . Only the first job completes on-time in schedule  $\pi$ . However, in an adversarial schedule  $\pi' = (2, 3, 1)$ , jobs 2 and 3 complete on-time, while only job 1 is late, giving the regret value 1. As shown in the example, for a given solution there may be infinitely many worst-case scenarios.

For any fixed schedule  $\pi$  we can write a mixed-integer linear program (MIP), which allows to compute the worst-case processing times, as well as the value of maximum regret. The program is the following:

$$\text{maximize } \sum_{j \in J} w_j (z_j - q_j), \tag{3}$$

subject to:

$$\sum_{j \in J} v_j \leq d, \tag{4}$$

$$\forall_{k=1, \dots, n} \sum_{i=1}^k p_{\pi(i)} + d_\epsilon q_{\pi(k)} \geq d_\epsilon, \tag{5}$$

$$\forall_{j \in J} v_j - p_j^+ z_j \leq 0, \tag{6}$$

$$\forall_{j \in J} p_j + p_j^+ z_j - v_j \leq p_j^+, \tag{7}$$

$$\forall_{j \in J} v_j - p_j \leq 0, \tag{8}$$

$$\forall_{j \in J} p_j^- \leq p_j \leq p_j^+, \tag{9}$$

$$\forall_{j \in J} z_j \in \{0, 1\}, q_j \in \{0, 1\}. \tag{10}$$

Binary decision variable  $z_j$  assumes value 1 if and only if job  $j$  is on-time in an adversarial schedule is the worst-case scenario, and binary decision variable  $q_j$  assumes value 1 if and only if job  $j$  is on-time in  $\pi$  in the worst-case scenario. Decision variable  $p_j$  represents the worst-case processing time of  $j$ th job. Values of these variables are determined through the set of constraints (5). These constraints are satisfied when  $q_{\pi(k)} = 0$ , for such jobs  $k$  that are on-time in  $\pi$  in the worst-case scenario,

and for  $q_{\pi(k)} = 1$  for such jobs  $k$  that are late. Constant  $d_\epsilon = d + \epsilon$  in (5), where  $\epsilon$  is a small positive value. Continuous variables  $v_j$  are introduced to linearize the mixed terms  $v_j = p_j z_j$ , through the set of constraints (6)–(8), as required for the constraint (4) to be linear.

Note that although standard solution algorithms for this program may require time increasing exponentially in  $n$ , in practice it can be solved very quickly. Computational experiments indicate, for example, that for  $n = 100$  optimal solutions can be computed in about one second on a modern computer, while even for thousands of jobs optimal solutions can be found within few minutes.

### 4 Finding Robust Solutions

We present a heuristic method that allows to determine solutions with low maximum regret for the problem (2). The method consists of two phases. In the first phase we try to determine a good initial solution, and in the second phase we use randomized local search in order to improve the initial solution.

The first phase is accomplished by solving a mixed-integer linear program that approximates the value of optimal robust solution. Let us consider a fixed schedule given by a permutation matrix  $\mathbf{x}$ . Since the optimization direction for robust schedule is the minimization, as opposed to the subproblem of maximization of regret (1), we form a dual program of the linear programming relaxation of (3)–(10). After relaxing (10) to  $0 \leq z_j \leq 1$  and  $0 \leq q_j \leq 1$ , for all  $j \in J$ , we can write:

$$\text{minimize } \sum_{j \in J} \left( -d_\epsilon \lambda_j^a + p_j^+ \lambda_j^b - p_j^- \lambda_j^c + \lambda_j^d + \lambda_j^e + p_j^+ \lambda_j^h \right) + d \lambda_0 \tag{11}$$

subject to:

$$\forall_{j \in J} \quad - \sum_{k=1}^n \sum_{i=1}^k x_{ij} \lambda_k^a + \lambda_j^b - \lambda_j^c - \lambda_j^g + \lambda_j^h \geq 0, \tag{12}$$

$$\forall_{j \in J} \quad - d_\epsilon \sum_{k=1}^n x_{kj} \lambda_k^a + \lambda_j^d \geq -w_j, \tag{13}$$

$$\forall_{j \in J} \quad \lambda_j^e - p_j^+ \lambda_j^f + p_j^+ \lambda_j^h \geq w_j, \tag{14}$$

$$\forall_{j \in J} \quad \lambda_j^f + \lambda_j^g - \lambda_j^h + \lambda_0 \geq 0. \tag{15}$$

Dual variable  $\lambda_0$  corresponds to the constraint (4), while the subsequent sets of dual variables  $\lambda^a$  through  $\lambda^h$  correspond to the constraints (5)–(10).

Since this is minimization program, we can also treat the matrix  $\mathbf{x}$  as a decision variable, and solve this program for unknown  $\mathbf{x}$ , along with  $\lambda$ , after adding the matching constraints:

$$\forall_{j \in J} \sum_{i=1}^n x_{ij} = 1, \tag{16}$$

$$\forall_{i=1, \dots, n} \sum_{j \in J} x_{ij} = 1, \tag{17}$$

$$x_{ij} \in \{0, 1\}. \tag{18}$$

Observe that in this case constraints (12) and (13) contain products of decision variables  $x_{ij}$  and  $\lambda_k^a$ . However, since  $x_{ij}$  are binary, and  $\lambda_k^a$  are nonnegative continuous, we can linearize these products in a standard way, by substituting new variables  $u_{kij} = \lambda_k^a x_{ij}$ , and adding three sets of constraints, similar to (6)–(8).

An optimal solution  $\mathbf{x}$  of (11)–(18) corresponds to an adversarial solution with fractional values of  $z_j$  and  $q_j$  (these are dual variables corresponding to (13)–(14)). In result, we get an upper bound on the optimal solution. This solution can be sometimes easily improved by rounding  $z_j$  and  $q_j$  to 0-1 values, and determining the corresponding  $\mathbf{x}$  that satisfies (4)–(10). We use the resulting binary matrix  $\mathbf{x}$  as an initial solution passed to the second phase of the method.

In the second phase, we apply a randomized local search heuristic. Given a permutation  $\pi$ , represented by a binary matrix  $\mathbf{x}$ , we compute the maximum regret  $Z(\mathbf{x})$  using program (3)–(10). In consecutive iterations, we swap two randomly selected jobs in  $\pi$ , obtaining a permutation  $\pi'$ , and compute the corresponding maximum regret  $Z(\pi')$ . Keeping track of the lowest value of maximum regret encountered so far, we either repeat the procedure by swapping the next pair of randomly selected jobs, if the new value is no higher than the current one, or otherwise we retract to the previous permutation  $\pi$ , by returning the previously swapped jobs to their previous positions.

The two-phase procedure can be summarized as follows:

1. (*phase 1*) Solve the mixed-integer program (11)–(18), obtaining fractional  $\tilde{\mathbf{z}}$  and  $\tilde{\mathbf{q}}$ , and binary  $\mathbf{x}_0$ .
2. Repeat for  $M$  iterations:
  - a. Round  $z_j = 1$  with probability  $\tilde{z}_j$ , and  $q_j = 1$  with probability  $\tilde{q}_j$ .
  - b. For binary  $\mathbf{z}$  and  $\mathbf{q}$  determine  $\mathbf{x}$  feasible for the set of constraints (4)–(10).
  - c. If  $Z(\mathbf{x}) < Z(\mathbf{x}_0)$  then put  $\mathbf{x}_0 \leftarrow \mathbf{x}$ .
3. (*phase 2*) Let  $\pi$  be a permutation corresponding to  $\mathbf{x}_0$ . Let  $S = \{\pi\}$  and  $\pi^* \leftarrow \pi$ .
4. Repeat for  $N$  iterations:
  - a. Create a schedule  $\pi'$  by swapping two randomly selected jobs  $i, j$  in  $\pi$ :  $\pi'(i) = \pi(j)$ ,  $\pi'(j) = \pi(i)$ , and  $\pi'(k) = \pi(k)$  for all  $k \neq i, j$ .
  - b. If  $\pi' \in S$  then discard  $\pi'$  and repeat the above step by taking another pair of random  $i, j$ . Otherwise,  $S \leftarrow S \cup \{\pi'\}$ .



- c. If  $Z(\pi') < Z(\pi)$  then  $\pi^* \leftarrow \pi'$ .
- d. If  $Z(\pi') \leq Z(\pi)$  then  $\pi \leftarrow \pi'$ . Otherwise, generate a random real number  $r \in [0, 1]$ . If  $r > \alpha$ , then  $\pi \leftarrow \pi'$ .

5. Return the schedule  $\pi^*$ .

The set  $S$  is maintained in order to prevent cycling during the search. The parameter  $\alpha \in [0, 1]$  controls the likelihood of proceeding from a worse than previous solution on the search path, and is intended to help avoiding local minima. This procedure can be run for prespecified number of iterations  $N$ , depending on the available computer resources, and can be easily parallelized. Note, however that for large number of jobs, as  $N \ll n!$ , this methods examines only a very small fraction of the search space.

## 5 Experimental Results

We have examined the solution technique presented in the previous section by comparing it with a simple mid-point heuristic, which is a standard method for tackling min-max regret problems with interval uncertainty [7]. This heuristic outputs a solution of the deterministic counterpart problem with a scenario fixed to interval middle points,  $\tilde{p}_i = p_i^- + \frac{1}{2}(p_i^+ - p_i^-)$ . Note that for the problem variant with arbitrary weights, this requires solving a knapsack problem.

In each experiment we have generated 10 problem instances for each value of the number of jobs  $n$ . Each such instance consisted of jobs with processing time intervals generated by taking the lower bound  $p_j^-$  as an uniformly random integer between 5 and 10, and the upper bound  $p_j^+$  by adding to  $p_j^-$  and uniformly random integer between 0 and 20. Due-dates were uniformly random integers between  $5n$  and  $10n$ . We have considered both unweighted ( $w_j = 1$  for all  $j \in J$ ) and weighted cases. In the latter, weights are uniformly random integers between 1 and 100.

The MIPs used by the solution method were implemented in CPLEX 12.6 software. For larger problem instances the program (11)–(18) in the phase 1 was usually not solved to optimality; instead, the best feasible solution was returned after running the solver for 60 s. However, for all the considered problem instances, program (3)–(10) was solved to optimality for every fixed permutation.

For each experiment we report the mean value and the standard deviation of the objective function, estimated from 10 problem instances. Values for both the mid-point scenario heuristic and the proposed method are given. We also report the computation time statistics for our method. Note that these depend on parameters that we have set:  $M = 100$  in step 2,  $N = 1000$  and  $\alpha = 0.1$  in step 4.

The results are presented in Tables 1 and 2. We conclude that the proposed method is consistently better than the mid-point scenario heuristic, especially for the variant of the problem with arbitrary weights.

**Table 1** Scheduling with the objective to minimize the (unweighted) number of late jobs

<i>n</i>	Mid-point heuristic		Proposed method				
	Mean Z	Std Z	Mean Z	Std Z	Min time	Mean time	Max time
10	2.90	0.30	2.90	0.30	73.73	108.95	164.79
15	3.70	1.00	3.40	1.11	200.48	418.10	618.06
20	4.70	1.10	4.50	1.57	296.33	408.98	539.51
25	5.80	1.24	5.20	2.04	509.21	574.94	632.99
30	5.70	0.90	5.60	1.11	533.80	644.97	679.01

**Table 2** Scheduling with the objective to minimize the total weight of late jobs

<i>n</i>	Mid-point heuristic		Proposed method				
	Mean Z	Std Z	Mean Z	Std Z	Min time	Mean time	Max time
10	140.90	34.65	94.00	33.74	66.46	77.06	101.70
15	195.00	47.04	111.80	50.24	187.39	419.93	886.15
20	243.20	67.79	142.56	72.19	281.25	564.59	637.19
25	464.50	118.42	155.50	80.31	668.37	728.07	850.91
30	444.80	113.36	149.67	58.38	695.58	914.52	1132.68

## 6 Conclusions

Single machine scheduling to minimize the total weight of late jobs with arbitrary processing times and a common due-date is an example of combinatorial problem which is easy to solve if exact values of parameters are known. In practice, this assumption is rarely valid. It turns out that interval data min-max regret variant of this problem is much more difficult to solve to optimality. We have examined a MIP-based heuristic solution method that successfully handles medium-sized problem instances, and appears to significantly improve on the standard mid-point heuristic. One of the future research directions is the design of efficient approximation methods for the class of problems in question.

## References

1. Aissi, H., Bazgan, C., Vanderpooten, D.: Min-max and min-max regret versions of combinatorial optimization problems: a survey. *Eur. J. Operat. Res.* **197**(2), 427–438 (2009)
2. Ali Aloulou, M., Federico Della, C.: Complexity of single machine scheduling problems under scenario-based uncertainty. *Operat. Res. Lett.* **36**(3), 338–342 (2008)
3. Aharon, B.-T., Laurent, E.G., Arkadi, N.: *Robust optimization*. Princeton University Press, (2009)
4. Brucker, P.: *Scheduling algorithms*. Springer, (2007)

5. Drwal, M.: Minimizing the weighted number of late jobs with interval processing times uncertainty. (Submitted for publication 2017)
6. Drwal, M., Rischke, R.: Complexity of interval minmax regret scheduling on parallel identical machines with total completion time criterion. *Operat. Res. Lett.* **44**(3), 354–358 (2016)
7. Goerigk, M., Anita, S.: Algorithm engineering in robust optimization. In: *Algorithm Engineering*, pp. 245–279. Springer, (2016)
8. Kasperski, A., Zielinski, P.: Minmax (regret) scheduling problems. In: *Sequencing and Scheduling with Inaccurate Data*. (eds.) Sotskov, Y., Werner, F., pp. 159–210, 2014
9. Milnor, J.: Games against nature. Technical report, DTIC Document (1951)

# Practical Solution to Parallel Machine Scheduling Problems

E. Parra

**Abstract** Parallel machine scheduling problems, even medium-sized ones, are very time consuming. The time taken to find a solution is very often more important than the exact solution itself. Different objective optimization functions depend on the specific situation. This paper presents a mathematical model and the software to implement it. The model allows users to switch between different objective functions and select the accuracy of the solution: from the fastest solution to the most exact one. It can be used with equal or different processing times for each job in different machines.

**Keywords** Scheduling • Parallel machines • Optimization models

## 1 Introduction

Parallel machine scheduling has been studied extensively over the last 50 years and is summarized in different surveys [6]. In general terms, the problem can be described as follows. A set of  $n$  jobs  $J = \{J_1, J_2, \dots, J_n\}$  must be scheduled on  $m$  parallel machines  $M = \{M_1, M_2, \dots, M_m\}$ . Job  $J_i$  has a processing requirement  $p_i$ , and machine  $M_j$  operates at a speed  $v_{i,j}$  when processing job  $J_i$ . The time it takes for job  $J_i$  to be processed by machine  $M_j$  is  $p_i/v_{i,j}$ . If  $v_{i,j} = 1$  for all  $i$  and  $j$ , then the machines are referred to as identical machines. If  $v_{i,j} = v_j$  for all  $i$ , then the machines are referred to as uniform machines. Finally, if  $v_{i,j}$  is totally arbitrary, then the machines are referred to as unrelated machines. According to the 3-field notation introduced by Graham et al. [4], P, Q and R are used to denote identical, uniform and unrelated machines respectively. The goal is to schedule these  $n$  jobs on the  $m$  machines so as to optimize a given objective function.

---

E. Parra (✉)

Department of Economics, University of Alcalá, Madrid, Spain  
e-mail: enrique.parra@uah.es

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_62

621

Numerous objective functions have been proposed and studied in the literature. The most common include the makespan (the time it takes to complete all jobs), the total weighted completion time, number of tardy jobs, maximum weighted tardiness and total weighted tardiness, among others.

Recently, parallel machine scheduling has been studied under machine eligibility constraints. In this type of model, job  $J_i$  cannot be processed on just one of the  $m$  machines, but only on a machine belonging to a specific subset of the  $m$  machines, namely subset  $M_i \subseteq M$ , which is designated the processing set of job  $J_i$ . This problem is known as parallel machine scheduling subject to machine eligibility constraints.

A wide variety of potential situations may arise depending on the industry. Although equal processing time is justified in many scheduling situations, this is not always the case. Kravchenko and Werner [5] examined scheduling with equal processing times, and Brucker and Kravchenko [1] studied due date assignments with equal processing times. The paper by Lee et al. [6] reviewed makespan minimization scheduling optimization with machine eligibility. Shabtay et al. [8] studied offline scheduling with rejection. Other recent results in the field are Correa and Wagner [2] and Mohabbati-Kalejahi and Yoon [7] and Tavares et al. [9].

This paper introduces a software to solve the unrelated parallel machine scheduling problem in a general form (including P, Q or R types), with the possibility of mixing several objective functions, as occurs in practical situations: minimising makespan, minimising the total cost of processing the jobs, assuming it is not the same in every machine, minimising the total cost of processing the jobs assuming it is not the same in every machine plus the waiting cost (if any), and other variations.

To summarize its characteristics, it features a combination of different objective functions, different processing times and some types of constraints: machine eligibility constraints, jobs not arriving at the same time, jobs needing to be processed at a predefined fixed time. All of these characteristics may be present in the real world and often occur with limited decision time (online versus offline). The model and the procedure presented here allow a feasible solution to be obtained very quickly, and this solution can be refined later if necessary. The most useful way to use this model is by connecting it to the organization's internal data so it can rapidly adapt to the new circumstances.

## 2 Model Formulation

Before looking at the mathematical formulation in detail, we describe the model fundamentals.

## 2.1 Model Description

A set of  $n$  jobs ( $J_i$ ) with integer processing times  $p_{ij}$  must be processed in one of the  $m$  machines  $M_j$  available. It is possible that a specific job cannot be processed in a specific machine (machine eligibility constraints). The goal is to determine the starting time of each job and which machine to process it in order to minimize a particular objective function based on costs or processing times. Situations sometimes arise in real life that involve a waiting cost, or an additional cost for finishing the job after a predefined time. The processing cost and processing time is different for each machine depending on the particular job.

The proposed model has built-in adjustable accuracy; i.e. the model is able to obtain exact solutions, but if the resolution time is the priority, the model can be set to prioritize time over absolute accuracy. A parameter controls the accuracy: less accuracy generates a simpler model that can be solved faster, while the exact solution requires a larger model and more processing time to obtain the absolute optimal solution. This approach is valid for a wide range of objectives in the machine scheduling world. Minimising a cost function may entail several terms: sum of processing time of all the jobs, sum of processing costs, sum of cost proportional to due date or sum of delay costs from a time at which the costs become accountable (different from the arrival time or process start time).

The known data are: job arrival date, cost starting date (the time at which cost must be considered), job operating time in each machine, available machines, machine ready time. We will assume that each job is processed until its completion once it has been started.

The model is based on discrete variable time: a job  $i$  needs a processing time  $p'_{ij}$  (say minutes) to be completed at machine  $j$ . We will define  $h$  (minutes) as the length of a "period". So,  $p'_{ij}$  time is converted to a  $p_{ij}$  number of periods; these numbers have been calculated as the next integer obtained by rounding up the  $p'_{ij}/h$  ratio. This procedure creates a new time scale in such a way that the time horizon is converted to a number of  $T$  periods depending on the  $h$  value. If  $h$  increases,  $T$  decreases. The problem is therefore to determine in which period  $k$  ( $k = 1, 2, \dots, T$ ) and in which machine  $j$  to begin to process job  $i$ . The job will leave the machine (no interruptions are allowed) at the end of period  $k + p_{ij}$ .

As mentioned before, both types of parameters ( $p_{ij}$  and  $T$ ) can be modified by either increasing or decreasing a single parameter  $h$ : higher values reduce the number of periods. If all  $p_{ij}$  are integer numbers and the value  $h = 1$  has been selected, the solution will be optimal, so the above procedure is exact. This also occurs when there is a maximum common divisor for all  $p_{ij}$ .

## 2.2 *Mathematical Model*

Once parameter  $h$  has been selected, the model is built with  $X_{ijk}$  binary variables: if the value is 1, job  $i$  is processed at machine  $j$  in time period  $k$ .

The following types of constraint are created: (1) Each job is processed exactly once; (2) Each job is processed at one only machine; (3) Each job begins processing in one period precisely; (4) A maximum of one job can be processed in each machine in each period; each machine is occupied from the start time of job  $i$  until  $p_{ij}$ —operating periods needed for job  $i$  at machine  $j$ —periods later; and (5). In certain circumstances a job can be processed in a predetermined time period (this is a real-world specification).

The objective function can be changed, but the main features of the model will remain the same. In this case, we propose minimising a weighted sum of total processing time, delay costs, operational costs of each job, and arrival time waiting costs.

Delay costs are counted from a different date from the arrival date, and operational costs are different for each job/machine pair, so the objective function includes many options. Depending on the relative costs of the four components there is scope to minimize operational costs, delay costs and others.

This offers a wide range of flexible options for the practical use of the model.

In summary, the model is as follows:

### (I) *Indexes*

- $i \in J$  (Jobs)
- $j \in M$  (Machines)
- $k, k' \in T$  (Periods)

### (II) *Data*

- $a_i$  Period when job  $i$  is ready to be processed
- $t_i$  Period when job  $i$  delay costs are counted
- $r_j$  Period when machine  $j$  is available
- $d_i$  Delay costs of job  $i$  (um/period) from  $t_i$
- $w_i$  Delay costs of job  $i$  (um/period) from  $a_i$
- $c_{ij}$  Cost index of job  $i$  operations at machine  $j$
- $p_{ij}$  No. of periods required by job  $i$  to complete operations at machine  $j$   
( $p_{ij} = 0$  indicates that job  $i$  cannot use machine  $j$ )
- $f_i$  Compulsory period for beginning processing of job  $i$
- $W_t$  Weight for processing times
- $W_d$  Weight for delay times
- $W_c$  Weight for processing costs
- $W_a$  Weight for delay costs

(III) *Binary variables:*  $X_{ijk} \in \{0,1\}$ ; ( $X_{ijk} = 1$  if  $J_i$  is processed at  $M_j$  in period  $k$ )  
 The model is built using a model-generating software and will only create the variables compatible with:  $k \geq a_i$ ,  $k \geq r_j$  and  $p_{ij} > 0$

(IV) *Constraints:*

$$\sum_j \sum_k x_{ijk} = 1 \quad \forall i \tag{1}$$

$$\sum_k x_{ijk} \leq 1 \quad \forall i, j / p_{ij} > 0 \tag{2}$$

$$\sum_i \sum_j x_{ijk} \leq 1 \quad \forall j, k' / k' > f_j, k' > k, k \geq [k' - p_{ij} + 1] \tag{3}$$

$$\sum_j x_{ijk} = 1 \quad \forall i, k / k = f_i \tag{4a}$$

$$\sum_j x_{ijk} \leq 1 \quad \forall i, k / k \neq f_i \tag{4b}$$

Constraints (1) ensure job  $i$  must be processed exactly once. Constraints (2) oblige each job/machine pair to be used in exactly one period or in none. Constraints (3) limit a job to be processed in one period and at one machine when this machine is occupied. Once job  $i$  begins processing in machine  $j$  (in period  $k$ ), it will occupy that machine during the next  $p_{ij}$  periods. It is therefore necessary to guarantee that no other job will begin during that time in that machine. This is done by creating an equation for each machine ( $j$ ) and period ( $k'$ ) pair. Each equation considers the sum of all the variables which, if their value were 1, would occupy the machine in the period  $k'$ . The sum is extended to all the jobs and periods between  $k'$  and the previous ( $k' - p_{ij}$ ). Constraints (4a) and (4b) are mutually exclusive: the first allows the process start time period to be chosen for job  $i$ ; the second requires the job to be processed beginning in period  $f_i$ . Both versions allow any machine to be selected (if it is compatible with the job) and requires that exactly one of them must be used.

(V) *Objective function:*

$$\sum_i \sum_j \sum_k [W_t A + W_c B + W_d C + W_a D] \cdot x_{ijk}$$

Where  $A = (k - 1 + p_{ij})$ ;  $B = c_{ij}$ ;  $C = d_i \cdot [k - p_{ij} - t_i]$  if  $t_i < k + p_{ij}$ ;  
 $D = w_i [k - a_i]$  if  $t_i \geq k + p_{ij}$ ,  $a_i < k$ ,  $a_i < t_i$

The objective function is the weighted sum of four terms: the first computes the sum of the processing times; the second, the sum of the machines' operating costs; the third assigns delay costs if job  $i$  is finished after date  $t_i$ ; and the last takes into account the delay cost from the arrival time ( $a_i$ ). More terms can be included, or some terms could be eliminated in a particular case. This is a flexible method for



practical management purposes: the model could then minimize the sum of processing times, processing costs, delay from different dates, delay from arrival or any combination of the four terms of the objective function.

If the cardinals of sets  $J$ ,  $M$  and  $T$  are  $n$ ,  $m$  and  $s$  respectively, the model described is made up comprises (in the worst case):  $n \cdot (s + m + 1) + s$  equations and  $(n) \cdot (m) \cdot (s)$  binary variables.

It should be noted that the most important result of the binary model solution is the assignment of each job to a machine and the optimal sequence in which each job will be processed in each machine. A post-optimization algorithm then establishes the exact times the jobs must be processed.

As stated before, the computer program (model builder) must generate exactly the variables and constraints needed. In practical situations this program is key to the success of the whole system. Once the binary problem has been resolved, the final practical solution is established with another computer program (post-optimizer procedure) based on the binary solution.

### 2.3 Complete Solution Procedure

The complete solution procedure implemented with the software modules is:

1. Select  $h$  parameter (i.e.  $h = 2$ )
2. *Model builder program*:
  - *Convert each processing time* (original numbers  $p'_{ij}$  to periods by rounding up. For example if  $p'_{12}$  (time of job 1 in machine 2) is 2, then  $p_{12} = 1$  if  $h = 2$ ; if  $p'_{23}$  (time of job 2 in machine 3) is 5, then  $p_{23} = 3$ . The exact solution is obtained if  $h = 1$  and all processing times are integers (that is, not restrictive for practical purposes).
  - *Formulate the binary problem and solve it*. Solution is in “periods”. If job 1 can be optimally processed in machine 2, beginning in period 3, then if  $h = 2$  has been selected, the due date will be the end of period 3.
3. *Optimization-solver*. The binary problem can be solved with any of the commercial software available (see [3]).
4. *Post-optimizer procedure*. This computer program calculates the exact time required to process the jobs using the optimal value of the binary variables, i.e. the job/machine assignments and the sequencing of jobs in each machine. For example, if  $p_{12}$  is 3 h and  $h = 2$ , this job will occupy 2 periods (so, 4 h instead of the 3 really necessary, which is a waste) so the following job would begin at  $t = 4$  (period 3 with  $h = 2$ ). This waste/inaccuracy is solved by the post-optimal computer software that adjusts to real times taking into account the processing order in each machine, respecting the sequencing and assignment of jobs (and

**Table 1** Computing time and accuracy

Trade-off time/accuracy	H1	H2	H3	H4	H6
Jobs	50	50	50	50	50
Machines	2	2	2	2	2
Horizon (days)	10	10	10	10	10
Hours per period	1	2	3	4	6
Number of periods	240	120	80	60	40
<b>Solution time scale</b>	<b>15.1</b>	<b>3.8</b>	<b>2.5</b>	<b>1.6</b>	<b>1.0</b>
<b>Error from exact solution (H1) (%)</b>	<b>0.0</b>	<b>0.5</b>	<b>0.9</b>	<b>2.0</b>	<b>3.6</b>

all the other constraints), but using real processing times. In the example, job 2 in the machine will begin at  $t = 3$  rather than at  $t = 4$ , and so on.

The data can be input with any system such as spreadsheet files (must include step 1). Computer programs for steps 2 and 4 can be coded with any programming language.

### 2.4 Experiment

Table 1 shows the different combinations of size, speed and accuracy as an example.

In the experiment with random processing times,  $n = 50$ ,  $m = 2$ . Different options were selected for the key parameter  $h$ . The exact solution was found with  $h = 1$  (case H1). With  $h = 2, 3, 4$  or  $6$  the solution is an approximation, but as shown in Table 1, it is very good for practical purposes. The maximum size of the binary problem is around 12,000 equations and 24,000 binary variables; but only 2000 and 4000 respectively if  $h = 6$  is chosen. H1 requires 15 times more computing time to solve than  $h = 6$ , which is only 3.6% worse; and 6 times more than  $h = 3$ , which has an error of less than 1%. Of course, the results depend on the particular problem.

## 3 Conclusions

The proposed method (mathematical model and solution procedure) allows flexibility in the accuracy of the solution depending on the requirements of a particular problem. The exact solution can be achieved with this method of solving unrelated parallel machine scheduling with different constraints and objective functions. The method is capable of obtaining solutions very quickly when speed is the top priority in the search for a highly accurate solution. It can be adapted to a wide range of objective function combinations, and offers a flexible way of solving these types of

problems by selecting varying degrees of accuracy versus computing time, depending on the particular needs.

**Acknowledgements** I would like to thank the anonymous reviewers for their valuable recommendations, which that have been considered in the final version and to Ms. Prudence Brooke-Turner for her revision of the English manuscript.

## References

1. Brucker, P., Kravchenko, S.A.: Scheduling jobs with equal processing times and time windows on identical parallel machines. *J. Sched.* **11**, 229–237 (2008)
2. Correa, J.R., Wagner, M.R.: LP-based online scheduling: from single to parallel machines. *Math. Progr.* **119**(1), 109–136 (2009)
3. Fourer, R.: Linear programming: software survey. *OR MS Today. INFORMS.* **42**, 3 (2015)
4. Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G.: Optimization and approximation in deterministic sequencing and scheduling: a survey. *Ann. Discret. Math.* **5**, 287–326 (1979)
5. Kravchenko, S.A., Werner, F.: Parallel machine problems with equal processing times: a survey. *J. Sched.* **14**(5), 435–444 (2011)
6. Lee, K., Leung, J.Y.-T., Pinedo, M.L.: Makespan minimization in online scheduling with machine eligibility. *Ann. Oper. Res.* **204**(1), 189–222 (2013)
7. Mohabbati-Kalejahi, N., Yoon, S.W.: Parallel machines scheduling problem for minimization of maximum lateness with sequence-dependent setup times. In: *IIE Annual Conference Proceedings*, pp. 837–847 (2015)
8. Shabtay, D., Gaspar, N., Kaspi, M.: A survey on offline scheduling with rejection. *J. Sched.* **16**(1), 3–28 (2013)
9. Tavares Neto, R.F., Godinho Filho, M., Da Silva, F.M.: An ant colony optimization approach for the parallel machine scheduling problem with outsourcing allowed. *J. Intell. Manuf.* **26**(3), 527–538 (2015)

# Decomposition and Feasibility Restoration for Cascaded Reservoir Management

Wim van Ackooij, Claudia D'Ambrosio and Raouia Taktak

**Abstract** We consider the Unit Commitment subproblem dedicated to hydro valley management, also known as Hydro Unit Commitment Problem (HUCP). The problem consists in finding an optimal hydro schedule for hydro valleys composed of head-dependent reservoirs for a short term period in which the electricity prices and the inflows are forecasted. We propose a Mixed Integer Linear Programming (MILP) formulation for the problem. Then, we solve it using a Lagrangian relaxation based decomposition combined with local branching used to restore feasibility. Preliminary computations show promising results.

**Keywords** Hydro unit commitment · Decomposition · Local branching

## 1 Introduction

A hydro valley consists of several interconnected reservoirs and each reservoir is associated with a hydro plant composed of hydro units. A unit can work as a turbine or a pump. The goal of the HUCP is to determine the start-up and shut-down slots for turbines and pumps and the operational level of each unit, in order to minimize the operational cost, under several technical and strategic constraints. This problem is very complex because of its mixed-integer nonlinear structure, uncertainties related to water inflows and electricity prices, as well as the size and shape of real valleys.

---

W. van Ackooij  
EDF R&D, OSIRIS, 7 Boulevard Gaspard Monge, 91120 Palaiseau, France  
e-mail: wim.van-ackooij@edf.fr

C. D'Ambrosio (✉)  
LIX CNRS (UMR7161), École Polytechnique route de Saclay,  
91128 Palaiseau Cedex, France  
e-mail: dambrosio@lix.polytechnique.fr

R. Taktak  
ISIMS & LT2S/CRNS, Technopole of Sfax, Ons City 3021, Sfax, Tunisia  
e-mail: raouia.taktak@gmail.com

© Springer International Publishing AG 2017

A. Sforza and C. Sterle (eds.), *Optimization and Decision Science: Methodologies and Applications*, Springer Proceedings in Mathematics & Statistics 217,  
DOI 10.1007/978-3-319-67308-0\_63

Moreover, there are computational limits because the HUCP has to be solved quickly. The reader is referred to [10] for a deeper description of the problem, its variants and main challenges. Note that the HUCP is a subproblem of the so-called unit commitment problem. For more information about the unit commitment problem see [6, 9]. We consider a deterministic context, that is when water inflows and electricity prices are supposed to be determined and forecasted with high accuracy in advance. Our contribution consists of two main phases. First, using Lagrangian relaxation, the HUCP is split into a sequence of smaller and easy-to-solve subproblems that are coordinated by a dual master program. Then, a local branching heuristic is used within a feasibility recovery phase. Our objective is to investigate to what extent this decomposition and the use of primal restoration through local branching constraints can be an effective tool for finding quickly good feasible solutions for the HUCP.

The paper is organized as follows. In the next section we describe a MILP formulation for the problem. In Sect. 3, we propose our decomposition method as well as the techniques and heuristic used to find a feasible solution. Section 4 will be devoted to presenting preliminary computational results. Finally, concluding remarks as well as future lines work are given in Sect. 5.

## 2 Mathematical Model

### 2.1 Notations

Our MILP is inspired from the one developed at EDF, a large utility company in France (see [11]). We consider a discretized time horizon. Let  $\mathcal{T} = \{1, \dots, T\}$  be the set of time periods. The time step is constant and denoted by  $\Delta t$  (in hours). Consider a hydro valley composed of a set of reservoirs interconnected by units (turbines or pumps). A valley can be seen as a directed graph  $G = (N, A)$ , where  $N$  represents the set of vertices corresponding to reservoirs, and  $A$  is the set of arcs corresponding to the possible paths of water between reservoirs. An element of  $A$  is a pair (origin, destination) in  $N \times N$ . For each  $a \in A$ ,  $D_a$  is the duration (in number of time periods) of the transport of water for path (arc)  $a$ . Given a reservoir  $n$ ,  $\mathcal{A}(n) = \{m : (m, n) \in A\}$  is the set of upstream reservoirs and  $\mathcal{F}(n) = \{m : (n, m) \in A\}$  is the set of downstream reservoirs. Turbines are numbered from 1 to  $N_T$  and pumps from 1 to  $N_P$ . Let  $\sigma_T : \{1, \dots, N_T\} \rightarrow A$  (resp.  $\sigma_P : \{1, \dots, N_P\} \rightarrow A$ ) be the function that associates with a turbine (resp. pump) the corresponding arc (path) between the reservoirs situated upstream and downstream the turbine/pump.

### 2.2 Decision Variables

We define two families of decision variables. We associate with each reservoir  $n \in N$  and each time period  $t \in \mathcal{T}$  a variable  $v_{nt}$  corresponding to the volume of reservoir  $n$  at  $t$  (expressed in  $m^3$ ). We also define variables  $x_{it}$  (resp.  $y_{it}$ ) correspond-

ing to the water flow turbined by (resp. going through) unit  $i \in \{1, \dots, N_T\}$  (resp.  $i \in \{1, \dots, N_p\}$ ) at time period  $t \in \mathcal{T}$ . Water flows are expressed in  $m^3/h$ .

For a generic hydro generator unit, the power output is generally expressed as a nonlinear function of the water flow and the water volume in the reservoir. In order to avoid dealing with nonlinearities and reduce the HUCP to a MILP, we assume no head effect and that each unit can be divided in a set of fictive groups. These groups, also known as *discrete operational points*, can be activated ones after the others respecting a predefined order in order to produce (or consume) electric energy. For each turbine  $i \in \{1, \dots, N_T\}$  (resp. pump  $i \in \{1, \dots, N_p\}$ ), the corresponding set of operational points are denoted by  $\mathcal{X}_{it}$  (resp.  $\widehat{\mathcal{X}}_{it}$ ). With each turbine  $i \in \{1, \dots, N_T\}$  (resp. pump  $i \in \{1, \dots, N_p\}$ ), each operational point  $j \in \mathcal{X}_{it}$  (resp.  $j \in \widehat{\mathcal{X}}_{it}$ ) and each period  $t \in \mathcal{T}$  we associate a binary variable  $e_{ijt}$  (resp.  $\hat{e}_{ijt}$ ) satisfying the following constraint  $e_{i1t} \geq e_{i2t} \geq \dots \geq e_{i|\mathcal{X}_{it}|t}$  (resp.  $\hat{e}_{i1t} \geq \hat{e}_{i2t} \geq \dots \geq \hat{e}_{i|\widehat{\mathcal{X}}_{it}|t}$ ). Using these binary variables, the flow of a turbine  $i \in \{1, \dots, N_T\}$  (resp. pump  $i \in \{1, \dots, N_p\}$ ) at period  $t$  can be written as the sum of the different flows of the operational points, i.e.

$$x_{it} = \sum_{j \in \mathcal{X}_{it}} e_{ijt} \Delta X_{ijt}, \quad \text{resp.} \quad y_{it} = \sum_{j \in \widehat{\mathcal{X}}_{it}} \hat{e}_{ijt} \widehat{\Delta X}_{ijt}, \quad (1)$$

and the generated (resp. consumed) power is given by

$$P_{it} = \sum_{j \in \mathcal{X}_{it}} e_{ijt} \Delta P_{ijt}, \quad \text{resp.} \quad P_{it} = \sum_{j \in \widehat{\mathcal{X}}_{it}} \hat{e}_{ijt} \widehat{\Delta P}_{ijt}, \quad (2)$$

where  $\Delta X_{ijt}$ ,  $\widehat{\Delta X}_{ijt}$ ,  $\Delta P_{ijt}$ , and  $\widehat{\Delta P}_{ijt}$  are known and assumed to be given.

### 2.3 Constraints

The HUCP is difficult to solve since several constraints with continuous and discrete variables exist, including hydraulic coupling and storage limits of the reservoirs. These constraints, as well as others, are described in the sequel.

#### Operational level's order constraint

We propose to model the discretization of the nonlinear efficiency curve using the incremental method (see [1]). The corresponding constraints are given by the following inequalities.

$$e_{i(j+1)t} \leq e_{ijt} \quad \forall i = 1, \dots, N_T, t \in \mathcal{T}, j \in \mathcal{X}_{it}. \quad (3)$$

Similarly the constraints for a pump are as follows.

$$\hat{e}_{i(j+1)t} \leq \hat{e}_{ijt} \quad \forall i = 1, \dots, N_p, t \in \mathcal{T}, j \in \widehat{\mathcal{X}}_{it}. \quad (4)$$

In (3) and (4), we assume  $e_{i(|\mathcal{X}_{it}|+1)t} = \hat{e}_{i(|\widehat{\mathcal{X}}_{it}|+1)t} = 0 \quad \forall i = 1, \dots, N_T, t \in \mathcal{T}$ .

**Flow stability constraint**

By these constraints, we forbid a sudden change of the flow between two steps of time. This ensures a flow stability when going through the different time steps. The flows should in fact be constant for at least one step time before increasing or decreasing. These constraints are respectively expressed for turbines and pumps by the following inequalities.

$$-1 \leq -e_{ij(t-1)} + e_{ij(t)} - e_{ij(t+1)} \leq 0 \quad \begin{matrix} \forall t = 2, \dots, T-1, \\ \forall i = 1, \dots, N_T, j \in \mathcal{X}_{it}. \end{matrix} \quad (5)$$

$$-1 \leq -\hat{e}_{ij(t-1)} + \hat{e}_{ij(t)} - \hat{e}_{ij(t+1)} \leq 0 \quad \begin{matrix} \forall t = 2, \dots, T-1, \\ \forall i = 1, \dots, N_P, j \in \widehat{\mathcal{X}}_{it}. \end{matrix} \quad (6)$$

**Forbidding simultaneously turbining and pumping**

Some turbines are reversible and can be used as pumps. In our mathematical model, this implies that it exists a function  $\mathbf{f} : \{1, \dots, N_T\} \times \{1, \dots, N_P\} \rightarrow \{0, 1\}$  that identifies couples (turbine  $i$ , pump  $k$ ),  $i = 1, \dots, N_T$  and  $k = 1, \dots, N_P$  corresponding to the same reversible unit, i.e.,  $\mathbf{f}(i, k) = 1$ . For such units, turbining and pumping cannot be done simultaneously. A slot of time (typically half an hour) is in fact required before switching from a turbine to a pump and viceversa. This is given by the following inequalities. For all  $t = 1, \dots, T-1$ , for all  $i = 1, \dots, N_T$  and  $k = 1, \dots, N_P$  such that  $\mathbf{f}(i, k) = 1$

$$0 \leq e_{il(t)} + \hat{e}_{k1(t)} \leq 1, \quad 0 \leq e_{il(t)} + \hat{e}_{k1(t+1)} \leq 1, \quad \text{and} \quad 0 \leq e_{il(t+1)} + \hat{e}_{k1(t)} \leq 1. \quad (7)$$

**Ramp up/down constraints**

These constraints express the fact that the absolute value of the water flows difference from one time step to the next has a bound. These constraints for a turbine and a pump respectively are given by the following inequalities.

For all  $t \in \mathcal{T}$  and  $i = 1, \dots, N_T$  (resp.  $i = 1, \dots, N_P$ )

$$-g_i \Delta_t \leq \sum_{j \in \mathcal{X}_{it}} \left( e_{ijt} \Delta X_{ijt} - e_{ij(t-1)} \Delta X_{ij(t-1)} \right) \leq \bar{g}_i \Delta_t, \quad (8)$$

$$-g_i \Delta_t \leq \sum_{j \in \widehat{\mathcal{X}}_{it}} \left( \hat{e}_{ijt} \widehat{\Delta X}_{ijt} - \hat{e}_{ij(t-1)} \widehat{\Delta X}_{ij(t-1)} \right) \leq \bar{g}_i \Delta_t, \quad (9)$$

where  $\underline{g}_i, \bar{g}_i$  are the downward and upward gradient in  $m^3/h^2$ , and  $\sum_{j \in \mathcal{X}_{it}} e_{ij0} \Delta X_{ij0} = x_{i0}$  (resp.  $\sum_{j \in \widehat{\mathcal{X}}_{it}} \hat{e}_{ij0} \widehat{\Delta X}_{ij0} = y_{i0}$ ) is known.

### Bounds on volumes

Each reservoir has an upper and lower bound for the stored water volume (water volume bounds). This is given by the following simple bounds. For all  $n \in N$  and  $t \in \mathcal{T}$ ,  $\underline{V}_{nt} \leq v_{nt} \leq \overline{V}_{nt}$ .

### Water flow constraints

The volume in a reservoir at the current time step is equal to the volume in the reservoir at the previous time step plus the inflows (external, due to turbining of uphill plants, or to pumping of downhill plants in the previous time steps) minus the outflows (external, due to turbining, or to pumping). This can be written as follows.

$$\begin{aligned}
 v_{nt} = & v_{n(t-1)} + s_{nt}\Delta t + \Delta t \sum_{m \in \mathcal{A}(n)} \sum_{i \in \sigma_T^{-1}[(m,n)]} \sum_{j \in \mathcal{X}_{i(t-D_{(m,n)})}} e_{ij(t-D_{(m,n)})} \Delta X_{ij(t-D_{(m,n)})} \\
 & - \Delta t \sum_{m \in \mathcal{F}(n)} \sum_{i \in \sigma_T^{-1}[(n,m)]} \sum_{j \in \mathcal{X}_i} e_{ijt} \Delta X_{ijt} \\
 & + \Delta t \sum_{m \in \mathcal{F}(n)} \sum_{i \in \sigma_P^{-1}[(m,n)]} \sum_{j \in \widehat{\mathcal{X}}_{i(t-D_{(m,n)})}} \hat{e}_{ij(t-D_{(m,n)})} \widehat{\Delta X}_{ij(t-D_{(m,n)})} \\
 & - \Delta t \sum_{m \in \mathcal{A}(n)} \sum_{i \in \sigma_P^{-1}[(n,m)]} \sum_{j \in \widehat{\mathcal{X}}_i} \hat{e}_{ijt} \widehat{\Delta X}_{ijt},
 \end{aligned} \tag{10}$$

where  $s_{nt}$  refers to the external inflows (e.g. from rain, rivers) for reservoir  $n \in N$  at time period  $t \in \mathcal{T}$ . As previously mentioned, these are supposed to be deterministic and forecasted for the next 2 days.

## 2.4 Objective Function

The objective is to minimize the following

$$- \sum_{n \in N} W_n v_{nT} - \Delta t \sum_{t \in \mathcal{T}} \lambda_t \sum_{i=1}^{N_T} \sum_{j \in \mathcal{X}_i} e_{ijt} \Delta P_{ijt} + \Delta t \sum_{t \in \mathcal{T}} \lambda_t \sum_{i=1}^{N_p} \sum_{j \in \widehat{\mathcal{X}}_i} \hat{e}_{ijt} \widehat{\Delta P}_{ijt}, \tag{11}$$

where  $W_n$  is the value of water savings for reservoir  $n \in N$ ,  $\lambda_t$  is the price signal (in €/MWh). The first term corresponds to the profit of water savings, the second part is the profit of the power generated by turbining, and the last term gives the cost of power consumption during pumping.

As it has been explained before, the HUCP presents many difficulties, namely related to its combinatorial structure and the flow constraints that link the different reservoirs. Therefore, we propose to solve this MILP using a Lagrangian relaxation (LR) technique combined with a local branching heuristic. The method is described in the following.



### 3 Proposed Method of Resolution

LR decomposition technique is based on three majors phases: dualization, dual problem solving, and primary recovery phase. The most common method is to dualize the linking constraints that constitute hard constraints expressing the physical relationship between upstream and downstream reservoirs. Another possible approach of dualization is the use of variables duplication, a strategy basically used for non-linear mixed integer programs [4]. After dualization, an important phase is to solve the dual subproblems and update the Lagrange multipliers. This can generally be handled by the use of nonsmooth algorithms like subgradient method [8]. Another interesting method to solve large-scale dual problems with high precisions is the bundle method [2], which helps giving good starting points for recovering primal solutions. The last step, after solving the dual problems, is to find a primal feasible solution.

In what follows, we will describe the methods and techniques that we have used in our algorithm for these three steps.

#### Dualization phase

An important feature of the LR approach is that it allows neglecting the complicating linking constraints for the HUCP which is consequently split into a sequence of smaller subproblems coordinated by a dual master program. To this end, we choose one (or many) reservoir(s) to cut the valley into zones. In the sequel, we assume, without loss of generality, that we will decompose the valley according to one reservoir, say  $r$ . The approach can be easily generalized in case of decomposition along many reservoirs. We write the dual problem with respect to flow conservation constraint (10) written for reservoir  $r$ . Let us denote by  $\mu$  the Lagrange multiplier associated with this constraint. Note that  $\mu$  is an element of  $\mathbb{R}^T$  and the Lagrangian is expressed as follows :

$$\begin{aligned} \mathcal{L}([\text{primal variables}], \mu) = & \\ & - \sum_{n \in N} W_n v_{nT} - \sum_{t \in \mathcal{T}} \lambda_t \Delta t \sum_{i=1}^{N_T} \sum_{j \in \mathcal{X}_{it}} e_{ijt} \Delta P_{ijt} + \sum_{t \in \mathcal{T}} \lambda_t \Delta t \sum_{i=1}^{N_p} \sum_{j \in \widehat{\mathcal{X}}_{it}} \hat{e}_{ijt} \widehat{\Delta P}_{ijt} \quad (12) \\ & + \sum_{t \in \mathcal{T}} \mu_t \mathcal{Q}(r, t), \end{aligned}$$

where  $\mathcal{Q}(r, t)$  is the flow constraint (10) written for reservoir  $r \in N$  at period  $t \in \mathcal{T}$ . The dual problem can be hence written

$$\max_{\mu \in \mathbb{R}^T} \Theta(\mu) \text{ where } \Theta(\mu) = \min_{[p.v.]} \mathcal{L}([p.v.], \mu). \quad (13)$$

Note here that  $\Theta$  is concave as it is the minimum of concave functions. Remark also that if  $\mu^*$  is a solution of (12), then  $\Theta(\mu^*)$  is a lower bound for the value of the primal problem.

**Dual problem solving**

Note that the Lagrangian (12) can be decomposed into independent parts each involving variables that are specific to the different zones obtained from the original valley after decomposition. In other words,  $\mathcal{L}([p.v.], \mu)$  can be seen as the sum of several independent subproblems. Consequently, using the simple fact that

$$\min_{x \in X, y \in Y} (f(x) + f(y)) = \min_{x \in X} f(x) + \min_{y \in Y} f(y),$$

computing  $\Theta$  reduces to solving separately the different independent subproblems obtained after decomposition. Moreover, as we know that  $\Theta$  is concave and we do not have an explicit expression of it, we can build a *cutting plane* model of it, see [7]. This idea is also exploited in bundle methods. After this cutting plane phase, one can compute an optimal pseudo-schedule for the HUCP. However, in the majority of cases, it is unfortunately not feasible. A recovery phase is then needed to restore feasibility, see, for example, [3].

**Primal feasibility and local branching**

We make use of the local branching heuristic [5] in order to restore feasibility of a given infeasible but integer solution. While the original local branching heuristic was designed to improve an integer feasible solution, here we use the local branching constraint to explore a limited neighborhood of an integer infeasible solution and restore feasibility. The infeasible starting solution is in general obtained by a rounding procedure applied on the optimal pseudo-schedule previously computed. This phase is based on the use of the following local branching constraint applied on a solution  $x^*$ .

$$\sum_{j \in S} x_j + \sum_{j \in \bar{J} \setminus S} (1 - x_j) \leq k, \tag{14}$$

where  $S = \{j \in \bar{J} : x_j^* = 0\}$ ,  $k$  is a given parameter, and  $\bar{J}$  is the set of binary variables.

**4 Preliminary Computational Results**

Our approach is developed in python, solved by Cplex v. 12.6.3, and tested on real-world instances provided by EDF. We test our approach in two phases. Starting from the integer infeasible solution  $x^*$ , we find the minimum value of  $k$  subject to (3)–(10) and (14) so as to find an integer feasible solution. After fixing the value of  $k$  to the value found in phase 1, we minimize (11) subject to (3)–(10) and (14).

Results are reported in Table 1. The first two columns give the instance number as well as the corresponding number of binary variables. The third column gives the minimum value of  $k$  to find a solution feasible for (3)–(10) and (14). The fourth and the last columns report the optimal solution value of the MILP (3)–(10) with

**Table 1** Results of the feasibility restoration algorithm on instances of increasing difficulty

Instance	# bin var	$k$ min	Sol (3)–(10) + (14)	Gap %	Sol (3)–(10)
1	121	0	7,892,201.81	0.00000000	7,892,201.81
2	242	0	7,897,761.96	0.00000000	7,897,761.96
3	363	0	7,903,420.33	0.00000000	7,903,420.33
4	484	0	7,908,672.93	0.00013340	7,908,683.48
5	605	0	7,913,848.94	0.00000000	7,913,848.94
6	726	0	7,918,722.82	0.00000000	7,918,722.82
7	847	0	7,923,505.37	0.00000000	7,923,505.37
8	968	2	7,927,457.86	0.00205610	7,927,620.86
9	1,089	8	7,930,990.53	0.01155535	7,931,907.09
10	1,210	2	7,935,081.02	0.00469765	7,935,453.80
11	1,331	3	7,938,807.36	0.00418673	7,939,139.75
12	1,452	1	7,942,466.53	0.01023444	7,943,279.48
24	2,904	3	7,995,659.39	0.00090173	7,995,731.49
48	5,808	6	8,099,578.24	0.01414516	8,100,724.10

and without the local branching constraint (14), respectively. And the second to last column reports the percentage GAP between the two solutions.

The results show that, for the smallest instances, i.e. 1–7, there is no need to change binary variables to find a feasible solution. Moreover, for all these instances but one, the feasible solutions found is also optimal. For the larger instances, the number of flips of the binary variable values needed to find a feasible solution is always less than 1% of the total number of binary variables. Moreover, the percentage GAP with respect to the optimal solution is always very limited, namely, less than 0.02%. Thus, the preliminary results show that the proposed method for finding effectively a high quality feasible solution for the HUCP is promising.

## 5 Conclusion

In this paper we presented a Lagrangian relaxation based decomposition method to solve the HUCP. The solution obtained after solving the dual problem is rounded to guarantee integrality, and feasibility is then ensured using a local branching constraint. Applied on real-world instances, our approach provides promising results. However, several improvements may be added in the future. We mainly aim at using bundle methods instead of the current cutting plane algorithm in order to ensure an efficient resolution of the dual problem. We also would like to include local branching not as a constraint but rather as a branching framework within a branch-and-bound algorithm. Moreover, extensive tests on real-world instances problems should be considered.

**Acknowledgements** This research benefited from the support of the FMJH Program Gaspard Monge in optimization and operation research and from the support to this program from EdF.

## References

1. Croxton, K.L., Gendron, B., Magnanti, T.L.: A comparison of mixed-integer programming models for nonconvex piecewise linear cost minimization problems. *Manag. Sci.* **49**(9), 1268–1273 (2003)
2. de Oliveira, W., Sagastizábal, C., Lemaréchal, C.: Convex proximal bundle methods in depth: a unified analysis for inexact oracles. *Mathe. Progr.* **148**(1), 241–277 (2014)
3. Dubost, L., Gonzalez, R., Lemaréchal, C.: A primal-proximal heuristic applied to french unit-commitment problem. *Mathe. Progr.* **104**(1), 129–151 (2005)
4. Finardi, E.C., Scuzziato, M.R.: A comparative analysis of different dual problems in the Lagrangian relaxation context for solving the hydro unit commitment problem. *Electr. Power Syst. Res.* **107**, 221–229 (2014)
5. Matteo, F., Andrea, L.: Local branching. *Math. progr.* **98**(1–3), 23–47 (2003)
6. Hechme-Doukopoulos, G., Brignol-Charousset, S., Malick, J., Lemaréchal, C.: The short-term electricity production management problem at edf. *Optima Newsl. Math. Optim. Soc.* **84**, 2–6 (2010)
7. Kelley Jr., J.E.: The cutting-plane method for solving convex programs. *J. Soc. Ind. Appl. Math.* **8**(4), 703–712 (1960)
8. Nilsson, O., Sjelvgren, D.: Variable splitting applied to modelling of start-up costs in short term hydro generation scheduling. *IEEE Trans. Power Syst.* **12**(2), 770–775 (1997)
9. Tahanan, M., van Ackooij, W., Frangioni, A., Lacalandra, F.: Large-scale unit commitment under uncertainty. *4OR* **13**(2):115–171 (2015)
10. Taktak, R., D'Ambrosio, C.: An overview on mathematical programming approaches for the deterministic unit commitment problem in hydro valleys. *Energy Syst.* **8**(1), 57–79 (2017)
11. van Ackooij, W., Bialecki, A., Triboulet, T.: La modélisation et mise en oeuvre du placement des vallées hydrauliques dans APOGENE v1.11. Technical report, EDF R&D, 2017

# Author Index

## A

Aghelinejad, Mohammad Mohsen, 591  
Allevi, Elisabetta, 403  
Amato, Flora, 31  
Ambrogio, Giuseppina, 257  
Amodeo, Lionel, 181  
Arbib, Claudio, 443  
Aringhieri, Roberto, 105, 113  
Avella, Pasquale, 443

## B

Bacci, Tiziano, 475  
Barbareschi, Mario, 31, 39  
Beraldi, Patrizia, 357  
Bertazzi, Luca, 507  
Bessi, Andrea, 195  
Bienstock, Daniel, 3  
Boccia, Maurizio, 93  
Boffa, Vincenzo, 195  
Bruni, Maria Elena, 357  
Bruno, Giuseppe, 245

## C

Caldarola, Enrico G., 83  
Capobianco, Giovanni, 517  
Carrabs, Francesco, 151, 529  
Carrozzino, Gianluca, 357  
Cerrone, Carmine, 151, 517, 539  
Cerulli, Raffaele, 367, 517, 529, 539  
Ciancio, Claudio, 257  
Clemente, Fabrizio, 139  
Conforti, Domenico, 121, 129  
Coniglio, Stefano, 287  
Cozzolino, Giovanni, 31

## D

D'Ambrosio, Ciriaco, 151, 367  
D'Ambrosio, Claudia, 629

Debertol, Daniele, 69  
De Causmaecker, Patrick, 13  
De Falco, Massimo, 375  
De Giovanni, Luigi, 161  
De Maio, Annarita, 507  
De Tommasi, Gianmaria, 305  
Diamantidis, Alexandros, 421  
Di Gangi, Massimo, 547  
Diglio, Antonio, 245, 601  
Di Pinto, Valerio, 83  
Di Puglia Pugliese, Luigi, 557, 577  
Drwal, Maciej, 611  
Duma, Davide, 105

## E

Esposito Amideo, Annunziata, 315, 567

## F

Faramondi, Luca, 315  
Felici, Giovanni, 517  
Festa, Paola, 529  
Fischetti, Martina, 203  
Fischetti, Matteo, 21  
Fliege, Jörg, 287  
Fonseca, Raquel J., 385  
Fraccaro, Marco, 203

## G

Gallo, Mariano, 267  
Galuzzi, Bruno, 171  
Garajová, Elif, 393  
Gastaldon, Nicola, 161  
Genovese, Andrea, 601  
Gentili, Monica, 367  
Gharote, Mangesh, 325  
Golden, Bruce, 539  
Groccia, Maria Carmela, 121  
Guerriero, Francesca, 557, 577

Guido, Rosita, [121](#), [129](#)

## H

Hladik, Milan, [335](#), [393](#)

## J

Jaime, Llorca, [275](#)

## K

Koukoumialos, Stelios, [421](#)

## L

Laganá, Demetrio, [257](#), [507](#), [577](#)

Landa, Paolo, [113](#)

Laureana, Federica, [529](#)

Lauriola, Ivano, [161](#)

Liberti, Leo, [213](#)

Lodha, Sachin, [325](#)

Lonati, Violetta, [223](#)

Lutz, Frederic, [181](#)

## M

Macrina, Giusy, [577](#)

Maggioni, Francesca, [403](#)

Malchiodi, Dario, [223](#)

Makkeh, Abdullah, [233](#)

Mancini, Simona, [113](#)

Marlière, Grégory, [495](#)

Marinelli, Fabrizio, [443](#), [453](#)

Marrone, Stefano, [49](#)

Masone, Adriano, [93](#)

Mastrandrea, Nicola, [375](#)

Mattia, Sara, [413](#), [443](#), [475](#)

Mazzeo, Antonino, [31](#)

Mazzocca, Nicola, [31](#)

Meda, Ermete, [69](#)

Mele, Adriano, [305](#)

Melisi, Alessia, [245](#)

Micillo, Rosario, [345](#)

Minnetti, Valentina, [59](#)

Mokalled, Hassan, [69](#)

Monga, Mattia, [223](#)

Montrone, Teresa, [485](#)

Morpurgo, Anna, [223](#)

## N

Nenni, Maria Elena, [345](#)

Nobili, Paolo, [485](#)

Ntio, Despoina, [421](#)

## O

Oliva, Gabriele, [315](#)

Ouazene, Yassine, [591](#)

## P

Papa, Cristina, [39](#)

Papi, Marco, [139](#)

Parra, E., [621](#)

Pascucci, Federica, [315](#)

Patil, Rahul, [325](#)

Pellegrini, Paola, [485](#), [495](#)

Pentangelo, Rosa, [539](#)

Presenti, Raffaele, [495](#)

Phuke, Nitin, [325](#)

Piantadosi, Gabriele, [49](#)

Piccolo, Carmela, [245](#), [601](#)

Pironti, Alfredo, [305](#)

Pizzuti, Andrea, [453](#)

Polimeni, Antonio, [547](#)

Pontecorvi, Luca, [139](#)

Pragliola, Concetta, [69](#)

## R

Rada, Miroslav, [393](#)

Raiconi, Andrea, [151](#)

Rarità, Luigi, [375](#)

Regoli, Fabio, [195](#)

Respen, Jean, [431](#)

Rinaldi, Antonio M., [83](#)

Rodriguez, Joaquin, [495](#)

Russo, Mauro, [461](#)

## S

Sansone, Carlo, [39](#), [49](#)

Sansone, Mario, [49](#)

Scaparra, Maria Paola, [315](#), [567](#)

Setola, Roberto, [139](#), [315](#)

Sforza, Antonio, [93](#), [275](#), [461](#)

Solina, Vittorio, [129](#)

Sottovia, Filippo, [161](#)

Sterle, Claudio, [93](#), [275](#), [461](#)

Stucchi, Eusebio, [171](#)

## T

Taktak, Raouia, [629](#)

Tammaro, Antonio, [31](#)

Tchoupo, Noubbissi M., [181](#)

Theis, Dirk Oliver, [233](#)

Tulino, Antonia Maria, [275](#)

## U

Ushakov, Anton, [295](#)

## V

van Ackooij, Wim, [629](#)

Vasilyev, Igor, [295](#)

Ventura, Paolo, [475](#)

Vidalis, Michael, [421](#)  
Vigo, Daniele, [195](#)  
Violi, Antonio, [357](#)

**W**

Walton, Ruth, [287](#)

**Y**

Yalaoui, Alice, [181](#), [591](#)  
Yalaoui, Farouk, [181](#)

**Z**

Zampieri, Elena, [171](#)  
Zufferey, Nicolas, [431](#)